ROBUST DISTRIBUTED OPTIMIZATION IN WIRELESS SENSOR NETWORK

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology

in

Telematics and Signal Processing

By

PYARI MOHAN PRADHAN

Roll No : 207EC103

Department of Electronics and Communication

National Institute of Technology

Rourkela, India

2009

ROBUST DISTRIBUTED OPTIMIZATION IN WIRELESS SENSOR NETWORK

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology

in

Telematics and Signal Processing

By

PYARI MOHAN PRADHAN

Under the Guidance of

Dr. G. Panda

Department of Electronics and Communication

National Institute of Technology

Rourkela, India

2009

National Institute of Technology

Rourkela

CERTIFICATE

This is to certify that the thesis entitled, **" Robust Distributed Optimization in Wireless Sensor Network "** submitted by **Pyari Mohan Pradhan** in partial fulfillment of the requirements for the award of Master of Technology Degree in Electronics & Communication Engineering with specialization in **Telematics and Signal Processing** during 2008 - 2009 at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Date

Prof. G. Panda (FNAE, FNASc)

Dept. of Electronics & Communication Engg.

National Institute of Technology

Rourkela-769008

Orissa, India

Acknowledgement

Upon completion of my M.Tech study, I would like to express my sincere gratitude to my advisor, teachers, seniors, parents and friends. They have given me enormous help in both study and life.

I would like to express my deep appreciation to my supervisor, **Prof. G. Panda** for his patience, encouragement, valuable guidance for my research. His creativity and passion for studies will always serve as an influential example for me. I sincerely appreciate the freedom Prof. G. Panda gave me in research and his countless insightful suggestions which will be beneficial for my life. It is him who led me to the research area of wireless sensor networks and guided me the way to high quality research work. His vast experience, patience and dedication were the strong pillars of my research career.

My cordial thanks to **Prof. Bernard Mulgrew**, Head of Institute of Digital Communication in University of Edinbergh for his collaboration and generous support at times when I needed most. His intelligence, great life attitude and passion for research truly inspired me. I would like to thank **Prof. B. K. Panigrahi** of IIT, Delhi for extending his help for my research work.

I would like to express my gratitude to research scholars **T. Panigrahi, U. K. Sahoo, A. K. Sahoo, S. K. Mishra, S. S. Sahu, B. Majhi and S. J. Nanda** for extending their help and support throughout the path of my research work. They provided insightful comments and suggestions on my research. I would also like to thank my friends **Nithin, Pavan, Vikas, Swanand, Somya and Sasmita** for their constant support and encouragement which helped me through the rough times of my research work.

Finally I would like to thank my parents for raising me in a way to believe that I can achieve anything in life with hard work and dedication.

Pyari Mohan Pradhan

Contents

Acknowledgement i

Contents ii

Abstract v

List of Figures vi

List of Abbreviations viii

1 Introduction 1

1.1 Introduction 2

1.2 Motivation 3

1.3 Thesis Layout 3

2 Distributed Incremental Least Mean Square Algorithm 5

2.1 Introduction 6

2.2 Incremental LMS Algorithm 6

2.2.1 Fundamental Concepts 6

2.3 Performance Analysis 7

2.3.1 Data Models and Assumptions 7

2.3.2 Weight-Energy Relation 9

2.3.3 Variance Relation 10

2.3.4 Gaussian Regressor 11

2.3.5 Diagonalization 12

2.3.6 Steady-State Behavior 13

2.4 Results and Discussion 15

2.5 Conclusion 17

3 Distributed Diffusion Least Mean Square Algorithm **18**
3.1 Introduction . . . 19
3.2 Diffusion LMS . . . 19
3.2.1 Diffusion LMS algorithm . . . 20
3.3 Network Global Model . . . 22
3.4 Performance Analysis . . . 23
3.4.1 Mean Transient Analysis . . . 24
3.4.2 Mean-Square Transient Analysis . . . 24
3.4.3 Steady State Analysis . . . 33
3.5 Results and Discussion . . . 34
3.6 Conclusion . . . 37

4 Distributed Block Least Mean Square Algorithm **38**
4.1 Introduction . . . 39
4.2 Block Diffusion LMS . . . 39
4.2.1 Diffusion Block LMS algorithm . . . 42
4.3 Network Global Model . . . 43
4.4 Performance Analysis . . . 44
4.4.1 Mean Transient Analysis . . . 44
4.4.2 Mean-Square Transient Analysis . . . 45
4.5 Results and Discussion . . . 48
4.5.1 Choice of Block Length . . . 49
4.5.2 Local Node performance . . . 50
4.6 Conclusion . . . 52

5 Robust Estimation in Wireless Sensor Network **53**
5.1 Introduction . . . 54
5.1.1 Distributed Optimization Techniques . . . 54
5.2 Decentralized Incremental Optimization . . . 55
5.3 Robust Estimation . . . 57
5.3.1 Robust incremental estimation during node failure . . . 58
5.3.2 Robust incremental estimation during node failure and impulsive noise condition . . . 60
5.4 Conclusion . . . 60

6 Robust Distributed Least Mean Square Algorithm **61**
6.1 Introduction . . . 62
6.2 Distributed Incremental LMS Algorithm with Error Saturation . . . 63
6.2.1 Adaptive Algorithm with Error Saturation Nonlinearity . . . 63
6.2.2 Model for Impulsive Noise . . . 64
6.2.3 Incremental LMS Algorithm with Error Saturation Nonlinearity . 64
6.3 Performance Analysis . . . 65
6.3.1 Data Models and Assumptions . . . 65
6.3.2 Weight-Energy Relation . . . 65
6.3.3 Variance Relation . . . 66
6.3.4 Evaluation of nonlinear term h_G and h_U . . . 68
6.3.5 Gaussian Regressor . . . 69
6.3.6 Steady-State Behavior . . . 69
6.4 Results and Discussion . . . 72
6.5 Conclusion . . . 75

7 Energy Efficient Layout for Wireless Sensor Network **78**
7.1 Introduction . . . 79
7.2 Related work . . . 79
7.3 Problem formulation . . . 80
7.3.1 WSN Modeling . . . 80
7.3.2 Calculation of Objectives . . . 80
7.4 Proposed MOPSO Algorithm . . . 83
7.5 Results and Discussion . . . 84
7.6 Conclusion . . . 86

8 Concluding Remarks and Future Work **88**
8.1 Conclusion . . . 89
8.2 Scope for Future Work . . . 90

Bibliography **91**

Abstract

Wireless sensor networks continue to get tremendous popularity, as evidenced by the increasing number of applications for these networks. The limiting factors of the sensor nodes, such as their finite energy supplies and their moderate processing abilities, as well as the unreliable wireless medium restrict the performance of wireless sensor networks. Energy efficient communication is a critical design objective for wireless sensor networks which are usually highly energy constrained. To achieve these goals, this thesis describes a distributed approach for solving several optimization problems in wireless sensor network.

The idea of distributed signal processing relies on the divide-and-conquer paradigm, which is often used in multiprocessor computers. According to the divide-and-conquer paradigm, a problem is divided into multiple sub-problems of smaller size. Every sensor solves each subproblem by using the same algorithm, and the solution to the original problem is obtained by combining the outputs from the different sensors. By designing appropriate communication protocols and collaborative computational schemes, sensors operate as distributed adaptive filters and generate the desired result. In an incremental mode of cooperation, information flows in a sequential manner from one node to the adjacent node. This mode of operation requires a cyclic pattern of collaboration among the nodes. In a diffusion implementation, on the other hand, each node communicates with all its neighbours as dictated by the network topology.

Often the objective of wireless sensor network is to get an estimate of some parameter or function of the data. In this case it may be beneficial to calculate these parameters in a distributed manner instead of sending the raw data to a central node for processing. This thesis investigates a class of distributed algorithms that circulate the estimates of parameters through the network. It uses theory of distributed incremental optimization to prove that the algorithm converges to the globally optimal value.

Layout is an important issue in designing sensor networks. This thesis describes a new approach for energy efficient layout of wireless sensor network. The sensors communicate with each other to transmit their data to a high energy communication node which acts as an interface between data processing unit and sensors. Optimization of sensor locations is essential to provide communication for a longer duration. It discusses an energy efficient layout with good coverage based on Multi-objective Particle Swarm Optimization algorithm.

List of Figures

2.1 (a) Regressor power profile. (b) Correlation index per node 15
2.2 (a) Noise power profile. (b) Signa-to-noise ration profile 16
2.3 (a) MSD Vs nodes for $\mu = 0.03$. (b) EMSE Vs nodes for $\mu = 0.03$ (c) MSE Vs nodes for $\mu = 0.03$ 16

3.1 Network Topology 35
3.2 Network statistics. (a) Network co-relation index pernode. (b) Regressor power profile. 35
3.3 Global mean-square deviation(MSD) curve for diffusion and non-cop. way of operation. 36
3.4 Global excess mean-square deviation(EMSE) curve for diffusion and non-cop. way of operation. 36
3.5 Local mean-square deviation(MSD) curve at node 1 36
3.6 Local mean-square deviation(MSD) curve at node 5 37
3.7 Local EMSE at nodes 1 and 5 for the same network 37

4.1 Network Topology 49
4.2 Network statistics. (a) Network co-relation index pernode. (b) Regressor power profile. 50
4.3 Global mean-square deviation(MSD) curve for diffusion and non-cop. way of operation. 50
4.4 Global excess mean-square deviation(EMSE) curve for diffusion and non-cop. way of operation. 51
4.5 Local mean-square deviation(MSD) comparison between block Diffusion LMS and Diffusion LMS. (a) MSD curve at node 1. (b) MSD curve at node 5. 51
4.6 Local EMSE at nodes 1 and 5 for the same network 52

5.1 Robust incremental estimation procedures during node failure 59
5.2 Robust incremental estimation procedures during node failure and impulsive noise condition 59

6.1 Regressor power profile 72
6.2 Correlation index per node 72
6.3 Noise power profile 73
6.4 Theoretical and simulated MSD Vs nodes curve for $p_r = 0.1$ 73
6.5 Theoretical and simulated EMSE Vs nodes curve for $p_r = 0.1$ 74
6.6 Theoretical and simulated EMSE Vs nodes curve for $p_r = 0.1$ 74
6.7 Theoretical and simulated MSD Vs nodes curve for $p_r = 0.5$ 75
6.8 Theoretical and simulated EMSE Vs nodes curve for $p_r = 0.5$ 75
6.9 Theoretical and simulated MSE Vs nodes curve for $p_r = 0.5$ 76
6.10 Theoretical and simulated MSD Vs step-size at node-7 for $p_r = 0.2$ . . . 76
6.11 Theoretical and simulated EMSE Vs step-size at node-7 for $p_r = 0.2$ . . . 77
6.12 Theoretical and simulated MSE Vs step-size at node-7 for $p_r = 0.2$ 77

7.1 Binary sensor coverage model 81
7.2 Stochastic sensor coverage model 81
7.3 Pareto front for a WSN with 10 sensors 85
7.4 Pareto-optimal layout with best coverage for a WSN with 10 sensors, 50 particles , 10 generations 85
7.5 Example of another pareto-optimal layout for a WSN with 10 sensors, 50 particles , 10 generations 86
7.6 Pareto fronts obtained with different sensor models for a a WSN with 10 sensors, 50 particles , 10 generations 86
7.7 Pareto fronts obtained for a WSN with 10 sensors and 50 particles 87

List of Abbreviations

BLMS	Block Least Mean Square
dAPA	distributed Affine Projection Algorithms
dNLMS	distributed Normalized Least Mean Square
EMSE	Excess Mean Square Error
LMF	Least Mean Fourth
LMS	Least Mean Square
MOEA	Multi-Objective Evolutionary Algorithm
MOO	Multi-Objective Optimization
MOPSO	Multi-Objective Particle Swarm Optimization
MSD	Mean Square Deviation
MSE	Mean Square Error
PSO	Particle Swarm Optimization
QoS	Quality of Service
RLS	Recursive Least Square

Chapter 1

Introduction

1.1 Introduction

Wireless Sensor Networks (WSN) are networks made up of tiny embedded devices. Each device is capable of sensing, processing and communicating the local information. The networks can be made up of hundreds or thousands of devices that work together to communicate the information that they obtain. Distributed processing deals with the extraction of information from data collected at nodes that are distributed over a geographic area. For example, each node in a network of nodes could collect noisy observations related to a certain parameter or phenomenon of interest. The nodes would then interact with their neighbours in a certain manner, as dictated by the network topology, in order to arrive at an estimate of the parameter or phenomenon of interest. The objective is to arrive at an estimate that is as accurate as the one that would be obtained if each node had access to the information across the entire network. In comparison, in a traditional centralized solution, the nodes in the network would collect observations and send them to a central location for processing. The central processor would then perform the required estimation tasks and broadcast the result back to the individual nodes. This mode of operation requires a powerful central processor, in addition to extensive amount of communication between the nodes and the processor. In the distributed solution, the nodes rely solely on their local data and on interactions with their immediate neighbours. The amount of processing and communication is significantly reduced.

In a WSN each node is responsible for covering a particular area by sensing. The node then sends the result to a sink node that collects the data. Nodes are used to relay the information, allowing the message to use multiple hops to reach the sink node. In order to process the information effectively, the network must have good coverage and the sink node must have good connectivity. Wireless Sensor Networks are frequently ad hoc, meaning that nodes can be added at any time and configure themselves to be part of the existing network. Any node can act as a relay to pass messages along the network. This works well for applications that add new sensors to replace those that have used up their battery life, or need to add more nodes for better coverage. Hence sensor placement needs to be done carefully considering the issues like coverage and connectivity.

Many sensor network design problems are characterized by the need to optimize multiple conflicting objectives. However, existing approaches generally focus on a single objective (ignoring the others), or combine multiple objectives into a single function to be optimized, to facilitate the application of classical optimization algorithms. This restricts their ability and constrains their usefulness to the network designer. A much

more appropriate and natural approach is to address multiple objectives simultaneously, applying recently developed multi-objective evolutionary algorithms (MOEAs) in solving sensor network design problems. This thesis describes and illustrates this approach by modeling a sensor network design problem (sensor placement), as a multi-objective optimization problem, developing the appropriate objective functions and discussing the trade-off between them.

1.2 Motivation

In wireless sensor network the fusion center provides a central point to estimate parameters. Energy efficiency, low latency, high estimation accuracy and fast convergence are important goals in estimation algorithms in sensor network. Depending on application and the resources, many algorithms exist to solve parameter estimation problem. One approach is the centralized approach which allows the most information to be present when making inference. However, the main drawback is the drainage of energy resources to transmit all observation to fusion center. Hence there was a need to find an approach that avoids the fusion center all together and allows the sensors to collaboratively make inference. This approach is called as the distributed scheme. Distributed computation of algorithms among sensors reduces energy consumption of the overall network, by trade-off between communication cost and computational cost. In order to make the inference procedure robust to nodal failure and impulsive noise, robust estimation procedure should be used.

Optimization of sensor locations in a network is essential to provide communication for a longer duration. In most cases sensor placement needs to be done in hostile areas without human involvement, e.g. by air deployment. The aircraft carrying the sensors has a limited payload, so it is impossible to randomly drop thousands of sensors over the ROI. Thus, the mission must be performed with a fixed number of sensors. The air deployment may introduce uncertainty in the final sensor positions. These limitations motivate the establishment of a planning system that optimize the WSN deployment process.

1.3 Thesis Layout

Chapter 2 gives an introduction to wireless sensor network and a new type of distributed cooperative strategy based on incremental technique. The individual nodes are trained

with LMS algorithm to estimate local parameters and share information with their immediate neighbour only. The steady state analysis of incremental LMS is shown both theoretically and using simulation.

Chapter 3 introduces a new type of adaptive distributed strategy based on diffusion technique. The individual nodes are equipped with local learning LMS algorithm to estimate local parameters and share information with their neighbors only. The steady state analysis of incremental LMS is shown both theoretically and using simulation.

Chapter 4 introduces a new type of block adaptive distributed strategy based on diffusion technique. The algorithm is distributed, cooperative like Incremental LMS and Diffusion LMS. Its performance is shown in terms of transient and mean-square error and is compared with non-cooperative schemes.

Chapter 5 proposes error saturation nonlinearity as a robust cost function. This is used for decentralized incremental estimation of parameters. The robustness of proposed estimation method is shown using simulation results and is compared with Huber loss function based estimation both in link failure and impulsive noise conditions.

Chapter 6 deals with the steady state analysis of distributed incremental LMS algorithm with error saturation nonlinearity. The performance measures for steady state are shown both theoretically and using simulation.

Chapter 7 deals with the modeling of sensor placement problem in a wireless sensor network. The sensor placement problem is modeled as a constrained multiobjective optimization problem that addresses multiple optimization criteria including the coverage and energy dissipated in the network.

Chapter 8 summarizes the results discussed in different chapters. Future work has also been discussed in brief.

Chapter 2

Distributed Incremental Least Mean Square Algorithm

2.1 Introduction

Networks consisting of nodes collecting data over a geographical area are envisioned to make a dramatic impact on a number of applications such as precision agriculture, disaster relief management, radar and acoustic source localization. In these applications, each node has some computational power, is able to send data to a subset of the network nodes, and tries to estimate a parameter of interest [1, 2]. Therefore, there is a great deal of effort in devising algorithms that are able to improve the estimate of the parameter of interest in every node with this information exchange between nodes [3, 4]. More precisely, in mathematical terms, each node should optimize a cost function that depends on the information available in the network.

The least mean square(LMS) algorithm is a popular adaptive algorithm because of its simplicity [5], [6]. The steady-state analysis of incremental LMS is shown here, both theoretically and using simulation. We have derived the performance equations by assuming that the input data is Gaussian and uncorrelated. Finally it is shown that the theoretical performance curves have excellent agreement with the corresponding simulation results.

Here we focus on real-valued data which can be extended for analysis of complex-valued data. Small bold letters are used to denote vectors, e.g., $\mathbf{w}$ denote the vector and capital bold letter e.g. $\mathbf{W}$ denotes the matrix. The symbol T denote transposition of vector. The notation $\|\mathbf{w}\|^2$ denotes the squared Euclidean norm of a vector $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w}$. Similarly $\|\mathbf{w}\|^2_\Sigma$ denotes the weighted-squared Euclidean norm $\|\mathbf{w}\|^2_\Sigma = \mathbf{w}^T\Sigma\mathbf{w}$. all vectors are column vector except for the input data vector denoted by $\mathbf{u}_i$, which is taken as row vector. The time instant is placed as a subscript for vectors and between parentheses for scalars, e.g. $\mathbf{w}_i$ and $e(i)$.

2.2 Incremental LMS Algorithm

2.2.1 Fundamental Concepts

The concept of incremental algorithms was developed in [7, 8, 9]. Consider a network having N number of nodes. Each node has access only to its immediate neighbor node. We assume that sensors make noisy vector measurements of their physical environment like local temperature, wind speed, humidity etc. Each node k has access to time realizations $\{d_k(i), \mathbf{u}_{k,i}\}$ of zero mean random data $\{d_k, \mathbf{u}_k\}, k = 1, 2, 3, ...N$, where $d_k(i)$ is a

scalar measurement and $\mathbf{u}_{k,i}$ is a $1 \times M$ regression row vector given as

$$\mathbf{u}_{k,i} = [u_k(i), u_k(i-1), \ldots u_k(i-M+1)]$$

Let $\Psi_k^{(i)}$ denote a local estimate of optimum weight $\mathbf{w}^\circ$ at node k at time i. Now assume that the node k has access to only $\Psi_{k-1}^{(i)}$, which is an estimate of $\mathbf{w}^\circ$ at its immediate neighbor node $k-1$ in the define topology. If at each time instant i we start with the initial condition $\Psi_0^{(i)} = \mathbf{w}_{i-1}$ at node 1 (*i.e.*, with the current global estimate $\mathbf{w}_{i-1}$ for $\mathbf{w}^\circ$), and iterate cyclicly across the nodes then, at the end of the procedure the local estimate at node N will be assigned to $\mathbf{w}_i$ *i.e.* $\mathbf{w}_i = \Psi_N^{(i)}$. Now the *distributed incremental LMS* algorithm [4] is defined as follows:
For each time $i \geq 0$, repeat:

$$\begin{aligned} k &= 1, \ldots, N \\ \Psi_0^{(i)} &= \mathbf{w}_{i-1} \\ \Psi_k^{(i)} &= \Psi_{k-1}^{(i)} + \mu_k \mathbf{u}_{k,i}^* \left(d_k(i) - \mathbf{u}_{k,i} \Psi_{k-1}^{(i)} \right) \\ \mathbf{w}_i &= \Psi_N^{(i)} \end{aligned} \tag{2.1}$$

2.3 Performance Analysis

Sayed [4] has shown an energy-based approach for the performance analysis of incremental LMS. The main objective of this type of analysis is, how close does each $\psi_k^{(i)}$(local estimate at node k in i-th data) get to the desired solution $\mathbf{w}^\circ$ as time evolves?

2.3.1 Data Models and Assumptions

The data model which is commonly used in literature of adaptive algorithms is used in our analysis. The desired unknown vector $\mathbf{w}^\circ$ relates to $\{d_k(i), \mathbf{u}_{k,i}\}$ as

$$d_k(i) = \mathbf{u}_{k,i}\mathbf{w}^\circ + v_k(i) \tag{2.2}$$

where $v_k(i)$ is a white Gaussian noise with variance $\sigma_{v,k}^2$ and independent of $\{d_k(i), \mathbf{u}_{k,i}\}$. It is assumed that the input data $u_k(i)$ to the nodes are spatially and temporally inde-

pendent. The local error signals at each node k are defined as

$$\tilde{\Psi}_k^{(i)} = \mathbf{w}^\circ - \Psi_k^{(i)} \quad \text{(weight error at time i)} \tag{2.3}$$

$$e_{a,k}(i) = \mathbf{u}_{k,i}\tilde{\Psi}_{k-1}^{(i)} \quad (a\ \textit{priori}\ \text{error}) \tag{2.4}$$

$$e_{p,k}(i) = \mathbf{u}_{k,i}\tilde{\Psi}_{k}^{(i)} \quad (a\ \textit{posteriori}\ \text{error}) \tag{2.5}$$

$$e_k(i) = d_k(i) - \mathbf{u}_{k,i}\tilde{\Psi}_{k-1}^{(i)} \quad \text{(output error)} \tag{2.6}$$

The output error $e_k(i)$ measures the estimation error in approximating $d_k(i)$ by using information available locally. By using the definition of *a priori* error $e_{a,k}(i)$ and the data model (2.2), the output error can be written as

$$e_k(i) = e_{a,k}(i) + v_k(i) \tag{2.7}$$

If we assume that the noise is independent of both weight and input data, then the variance relation can be written as

$$E\|e_k(i)\|^2 = E\|e_{a,k}\|^2 + E\|v_k(i)\|^2 = E\|e_{a,k}\|^2 + \sigma_{v,k}^2 \tag{2.8}$$

Our aim is to evaluate the steady-state values of the variances like mean-square error(MSE), excess-mean-square error(EMSE) and mean-square deviation(MSD) for every node which are the measures of performance of the filter. These quantities are defined as:

$$\eta_k = E\|\tilde{\Psi}_{k-1}^{(\infty)}\|^2 \quad (\mathbf{MSD}) \tag{2.9}$$

$$\zeta_k = E|e_{a,k}(\infty)|^2 \quad (\mathbf{EMSE}) \tag{2.10}$$

$$\xi_k = E|e_k(\infty)|^2 = \zeta_k + \sigma_{v,k}^2 \quad (\mathbf{MSE}) \tag{2.11}$$

Introducing the *weighted norm* notation for a vector x as $\|x\|_\Sigma^2 = x^*\Sigma x$ where $\Sigma(>0)$ is a Hermitian positive definite matrix. In order to study the steady state parameters we introduce [10] the weighted *a priori* and *a posteriori* error defined as

$$e_{a,k}^\Sigma(i) = \mathbf{u}_{k,i}\Sigma\tilde{\Psi}_{k-1}^{(i)} \qquad \text{and} \qquad e_{p,k}^\Sigma(i) = \mathbf{u}_{k,i}\Sigma\tilde{\Psi}_{k}^{(i)} \tag{2.12}$$

Later it will be shown that the different choices for Σ allows us to evaluate different performance measures. For $\Sigma = I$ we define

$$e_{a,k}(i) = \mathbf{u}_i \tilde{\Psi}_{k-1}^{(i)}, \qquad e_{p,k}(i) = \mathbf{u}_i \tilde{\Psi}_k^{(i)}$$

Combining (2.6) and (2.1) the incremental LMS can be written as

$$\Psi_k^{(i)} = \Psi_{k-1}^{(i)} + \mu_k \mathbf{u}_{k,i}^* e_k(i)$$

Subtracting $\mathbf{w}^\circ$ from both sides of above equation, we obtain

$$\tilde{\Psi}_k^{(i)} = \tilde{\Psi}_{k-1}^{(i)} - \mu_k \mathbf{u}_{k,i}^* e_k(i) \tag{2.13}$$

Relation between various error terms $e_{a,k}^{\Sigma}(i)$, $e_{p,k}^{\Sigma}(i)$ and $e_k(i)$ is obtained by premultiplying both sides of (2.13) by $\mathbf{u}_{k,i}\Sigma$ i.e.

$$\mathbf{u}_{k,i}\Sigma\tilde{\Psi}_k^{(i)} = \mathbf{u}_{k,i}\Sigma\tilde{\Psi}_{k-1}^{(i)} - \mu_k \|\mathbf{u}_{k,i}\|_{\Sigma}^2 e_k(i) \tag{2.14}$$

Using the definitions from (2.12)

$$e_{p,k}^{\Sigma}(i) = e_{a,k}^{\Sigma}(i) - \mu_k \|\mathbf{u}_{k,i}\|_{\Sigma}^2 e_k(i) \tag{2.15}$$

and,subsequently

$$e_k(i) = \frac{1}{\mu_k} \frac{\left(e_{a,k}^{\Sigma}(i) - e_{p,k}^{\Sigma}(i)\right)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \tag{2.16}$$

2.3.2 Weight-Energy Relation

Elimination of the error term $e(i)$ from (2.13) by using (2.15) we obtained

$$\tilde{\Psi}_k^{(i)} + \frac{\mathbf{u}_{k,i}^* e_{a,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} = \tilde{\Psi}_{k-1}^{(i)} + \frac{\mathbf{u}_{k,i}^* e_{p,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \tag{2.17}$$

Taking weighted energy on both sides of (2.17) and canceling the equal terms on both sides, we found

$$\|\tilde{\Psi}_k^{(i)}\|_{\Sigma}^2 + \frac{|e_{a,k}^{\Sigma}(i)|^2}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} = \|\tilde{\Psi}_{k-1}^{(i)}\|_{\Sigma}^2 + \frac{|e_{p,k}^{\Sigma}(i)|^2}{\|\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \tag{2.18}$$

The above equation is known as *space-time* weighted energy relation [4, 3].

2.3.3 Variance Relation

For compactness of notation the time index i is dropped.The variance relation can be obtained from the energy relation (2.18) by replacing a posteriori error by its equivalent expression from (2.15)

$$\|\tilde{\Psi}_k\|_{\Sigma}^2 = \|\tilde{\Psi}_{k-1}\|_{\Sigma}^2 - \mu_k e_{a,k}^{\Sigma *} e_k - \mu_k e_k^* e_{a,k}^{\Sigma} + \mu_k^2 \|\mathbf{u}_k\|_{\Sigma}^2 . |e_k^2| \tag{2.19}$$

Using (2.7) and taking expectation on both sides leads to

$$\begin{aligned} E\|\tilde{\Psi}_k\|_{\Sigma}^2 = E\|\tilde{\Psi}_{k-1}\|_{\Sigma}^2 - \mu_k E e_{a,k}^{\Sigma *} e_{a,k} - \mu_k E e_{a,k}^{\Sigma} e_{a,k}^* \\ + \mu_k^2 \sigma_{v,k}^2 E\|\mathbf{u}_k\|_{\Sigma}^2 + \mu_k^2 E\|\mathbf{u}_k\|_{\Sigma}^2 . |e_{a,k}^2| \end{aligned} \tag{2.20}$$

Using (2.12), expanding the above equation as follows:

$$\begin{aligned} E\|\tilde{\Psi}_k\|_{\Sigma}^2 = E\|\tilde{\Psi}_{k-1}\|_{\Sigma}^2 - \mu_k E\tilde{\Psi}_{k-1}^* \Sigma \mathbf{u}_k^* \mathbf{u}_k \tilde{\Psi}_{k-1} - \mu_k E\tilde{\Psi}_{k-1}^* \mathbf{u}_k^* \mathbf{u}_k \Sigma \tilde{\Psi}_{k-1} \\ + \mu_k^2 E\tilde{\Psi}_{k-1}^* \mathbf{u}_k^* \mathbf{u}_k \Sigma \mathbf{u}_k^* \mathbf{u}_k \tilde{\Psi}_{k-1} + \mu_k^2 \sigma_{v,k}^2 E\|\mathbf{u}_k\|_{\Sigma}^2 \end{aligned} \tag{2.21}$$

Given that $\|x\|_A^2 + \|x\|_B^2 = \|x\|_{A+B}^2$, the above equation can be written as

$$E\|\tilde{\Psi}_k\|_{\Sigma}^2 = E\left(\|\tilde{\Psi}_{k-1}\|_{\mathbf{\Sigma}'}^2\right) + \mu_k^2 \sigma_{v,k}^2 E\|\mathbf{u}_k\|_{\Sigma}^2 \tag{2.22}$$

in terms of stochastic weighting matrix

$$\mathbf{\Sigma}' = \Sigma - \mu_k(\mathbf{u}_k^* \mathbf{u}_k \Sigma + \Sigma \mathbf{u}_k^* \mathbf{u}_k) + \mu_k^2 \|\mathbf{u}_k\|_{\Sigma}^2 \mathbf{u}_k^* \mathbf{u}_k \tag{2.23}$$

Since regression data is completely independent, the first term in RHS of (2.22) can be written as

$$E\left(\|\tilde{\Psi}_{k-1}\|_{\mathbf{\Sigma}'}^2\right) = E\|\tilde{\Psi}_{k-1}\|_{E\mathbf{\Sigma}'}^2 \tag{2.24}$$

Hence (2.22) and (2.23) becomes

$$E\|\tilde{\Psi}_k\|_{\Sigma}^2 = E\|\tilde{\Psi}_{k-1}\|_{\Sigma'}^2 + \mu_k^2 \sigma_{v,k}^2 E\|\mathbf{u}_k\|_{\Sigma}^2 \tag{2.25}$$

where $\Sigma' = E\mathbf{\Sigma}'$ is given by

$$\Sigma' = \Sigma - \mu_k E(\mathbf{u}_k^* \mathbf{u}_k \Sigma + \Sigma \mathbf{u}_k^* \mathbf{u}_k) + \mu_k^2 E\|\mathbf{u}_k\|_\Sigma^2 \mathbf{u}_k^* \mathbf{u}_k \tag{2.26}$$

and Σ' is a *deterministic* matrix.

2.3.4 Gaussian Regressor

Assume that the regressors $\{\mathbf{u}_{k,i}\}$ arises from a circular Gaussian distribution with covariance matrix $\mathbf{R}_{u,k}$ which is defined as $\mathbf{R}_{u,k} = E[u_k^* u_k]$. Now introduce the eigen decomposition $\mathbf{R}_{u,k} = U_k \wedge_k U_k^*$, where U_k is unitary (*i.e.* $U_k^* U_K = U_k U_k^* = I$) and $\wedge_k$ is a diagonal matrix with eigenvalues of $\mathbf{R}_{u,k}$. Then define the transformed quantities as:

$$\overline{\Psi}_k = U_k^* \tilde{\Psi}_k \qquad \overline{\mathbf{u}}_k = \mathbf{u}_k U_k \qquad \overline{\Sigma} = U_k^* \Sigma U_k \qquad \overline{\Sigma}' = U_k^* \Sigma' U_k$$

Since U_K is unitary matrix, we have $\|\tilde{\Psi}_k\|_\Sigma^2 = \|\overline{\Psi}_k\|_{\overline{\Sigma}}^2$ and $\|\mathbf{u}_k\|_\Sigma^2 = \|\overline{\mathbf{u}}_k\|_{\overline{\Sigma}}^2$. Now under the change of variables, the variance relation (2.25) and (2.26) are written as

$$E\|\overline{\Psi}_k\|_{\overline{\Sigma}}^2 = E\|\tilde{\Psi}_{k-1}\|_{\overline{\Sigma}'}^2 + \mu_k^2 \sigma_{v,k}^2 E\|\overline{\mathbf{u}}_k\|_{\overline{\Sigma}}^2 \tag{2.27}$$

$$\overline{\Sigma}' = \overline{\Sigma} - \mu_k E(\overline{\mathbf{u}}_k^* \overline{\mathbf{u}}_k \overline{\Sigma} + \overline{\Sigma} \overline{\mathbf{u}}_k^* \overline{\mathbf{u}}_k) + \mu_k^2 E\|\overline{\mathbf{u}}_k\|_{\overline{\Sigma}}^2 \overline{\mathbf{u}}_k^* \overline{\mathbf{u}}_k \tag{2.28}$$

The first two moments we need to evaluate for the steady state analysis are straight forward and given as

$$E\|\overline{\mathbf{u}}_k\|_{\overline{\Sigma}}^2 = \mathrm{Tr}(\wedge_k \overline{\Sigma}) \quad \text{and} \quad E\overline{\mathbf{u}}_k^* \overline{\mathbf{u}}_k = \wedge_k \tag{2.29}$$

For Gaussian regressors the third moment is given as

$$E\|\overline{\mathbf{u}}_k\|_{\overline{\Sigma}}^2 \overline{\mathbf{u}}_k^* \overline{\mathbf{u}}_k = \wedge_k \mathrm{Tr}(\overline{\Sigma} \wedge_k) + \gamma \wedge_k \overline{\Sigma} \wedge_k \tag{2.30}$$

where $\gamma = 1$ for circular complex data and $\gamma = 2$ for real data. Substituting (2.29) and (2.30) into (2.27),(2.28) can be written as

$$E\|\overline{\Psi}_k\|_{\overline{\Sigma}}^2 = E\|\overline{\Psi}_{k-1}\|_{\overline{\Sigma}'}^2 + \mu_k^2 \sigma_{v,k}^2 \mathrm{Tr}(\wedge_k \overline{\Sigma}) \tag{2.31}$$

$$\overline{\Sigma}' = \overline{\Sigma} - \mu_k(\wedge_k\overline{\Sigma} + \overline{\Sigma}\wedge_k) + \mu_k^2(\wedge_k \mathrm{Tr}(\overline{\Sigma}\wedge_k) + \gamma \wedge_k \overline{\Sigma}\wedge_k) \tag{2.32}$$

2.3.5 Diagonalization

Since the choice of weighted matrix Σ is in our hand, let us choose Σ such that both $\overline{\Sigma}$ and $\overline{\Sigma}'$ will be diagonal in (2.32). Under this condition, it is possible to rewrite (2.32) in more compact form in terms of diagonal entries of $\{\overline{\Sigma}, \overline{\Sigma}'\}$. To do so, we define the vectors

$$\overline{\sigma} = \mathrm{diag}\{\overline{\Sigma}\}, \quad \overline{\sigma}' = \mathrm{diag}\{\overline{\Sigma}'\}, \quad \lambda_k = \mathrm{diag}\{\wedge_k\} \tag{2.33}$$

where $\{\overline{\sigma}, \overline{\sigma}', \lambda_k\}$ are $M \times 1$ column vectors with diagonal entries of the corresponding matrices. $\overline{\sigma}$ contains the diagonal entries of $\overline{\Sigma}$, where as λ_k contains the diagonal entries of $\wedge_k$ and $\overline{\sigma}'$ contains the diagonal entries of $\overline{\Sigma}'$. Here we use the notation diag$\{\}$ in two ways. For any arbitrary matrix A, $a = \mathrm{diag}\{A\}$, is a vector containing the main diagonal of A. For a column vector a, $A = \mathrm{diag}\{a\}$ results in a diagonal matrix whose entries are those of the vector a. Therefore, we can also write,

$$\overline{\Sigma} = \mathrm{diag}\{\overline{\sigma}\}, \quad \overline{\Sigma}' = \mathrm{diag}\{\overline{\sigma}'\}, \quad \wedge_k = \mathrm{diag}\{\lambda_k\}$$

in order to recover $\{\overline{\Sigma}, \overline{\Sigma}', \wedge_k\}$ from $\{\overline{\sigma}, \overline{\sigma}', \lambda_k\}$. Since $\mathrm{Tr}(\wedge_k\overline{\Sigma}) = \lambda_k^T\overline{\sigma}$, the expression (2.32) can be rewritten in terms of the vectors $\{\overline{\sigma}, \lambda_k\}$ as

$$\begin{aligned}\overline{\sigma}' &= (\mathrm{I} - 2\mu_k \wedge_k + \gamma\mu_k^2\wedge_k)\overline{\sigma} + \mu_k^2(\lambda_k^T\overline{\sigma})\lambda_k \\ &= \overline{F}_k\overline{\sigma}\end{aligned} \tag{2.34}$$

where the $M \times M$ coefficient matrix $\overline{F}_k$ is defined by

$$\overline{F}_k = \mathrm{I} - 2\mu_k \wedge_k + \gamma\mu_k^2 \wedge_k + \mu_k^2\lambda_k\lambda_k^T \tag{2.35}$$

We can rewrite (2.31) by using the vectors $\{\overline{\sigma}, \overline{\sigma}', \lambda_k\}$ instead of the matrices $\{\overline{\Sigma}, \overline{\Sigma}', \lambda_k\}$. Using (2.35) and the notation (2.33), (2.31) becomes

$$E\|\overline{\Psi}_k\|^2_{\mathrm{diag}\{\overline{\sigma}\}} = E\|\overline{\Psi}_{k-1}\|^2_{\mathrm{diag}\{\overline{F}_k\overline{\sigma}\}} + \mu_k^2\sigma_{v,k}^2(\lambda_k^T\overline{\sigma}) \tag{2.36}$$

For the sake of compactness, the diag{} notation will be dropped from the subscripts, keeping only the corresponding vectors

$$E\|\overline{\Psi}_k^{(i)}\|_{\overline{\sigma}_k}^2 = E\|\overline{\Psi}_{k-1}^{(i)}\|_{\overline{F}_k\overline{\sigma}_k}^2 + \mu_k^2\sigma_{v,k}^2(\lambda_k^T\overline{\sigma}) \tag{2.37}$$

where time index i is restored for clarity and $\{\overline{\sigma}, \overline{\sigma}'\}$ are replaced by $\{\overline{\sigma}_k, \overline{\sigma}'_k\}$ to indicate that the weighting matrix can be node dependent.

2.3.6 Steady-State Behavior

When $i \to \infty$, let us take $\overline{\mathbf{h}}_k = \overline{\Psi}_k^{(\infty)}$ and $\mathbf{g}_k = \mu_k^2\sigma_{v,k}^2\lambda_k^T$ as a row vector. Then for $i \to \infty$(i.e. in steady state), the variance relation (2.37) gives

$$E\|\overline{\mathbf{h}}_k\|_{\overline{\sigma}_k}^2 = E\|\overline{\mathbf{h}}_{k-1}\|_{\overline{F}_k\overline{\sigma}_k}^2 + \mathbf{g}_k\overline{\sigma}_k \tag{2.38}$$

The steady-state performances which are defined in (2.11) are now can measured by using (2.38),as follows:

$$\eta_k = E\|\overline{\mathbf{h}}_{k-1}\|_q^2, \quad q = \text{diag}\{I\} \quad (\textbf{MSD}) \tag{2.39a}$$

$$\zeta_k = E|\overline{\mathbf{h}}_{k-1}|_{\lambda_k}^2, \quad \lambda_k = \text{diag}\{\wedge_k\} \quad (\textbf{EMSE}) \tag{2.39b}$$

$$\xi_k = \zeta_k + \sigma_{v,k}^2 \quad (\textbf{MSE}) \tag{2.39c}$$

It is observed that (2.38) is a coupled equation where both $\overline{\mathbf{h}}_k$ and $\overline{\mathbf{h}}_{k-1}$ are involved which can be interpreted as information from two spatial locations. But the ring topology together with the weighting matrices can be exploited to resolve this difficulty. By iterating (2.38) we will get set of N coupled equalities as

$$\begin{aligned}
E\|\overline{\mathbf{h}}_1\|_{\overline{\sigma}_1}^2 &= E\|\overline{\mathbf{h}}_N\|_{\overline{F}_1\overline{\sigma}_1}^2 + \mathbf{g}_1\overline{\sigma}_1 \\
E\|\overline{\mathbf{h}}_2\|_{\overline{\sigma}_2}^2 &= E\|\overline{\mathbf{h}}_1\|_{\overline{F}_2\overline{\sigma}_2}^2 + \mathbf{g}_2\overline{\sigma}_2 \\
&\vdots \\
E\|\overline{\mathbf{h}}_{k-2}\|_{\overline{\sigma}_{k-2}}^2 &= E\|\overline{\mathbf{h}}_{k-3}\|_{\overline{F}_{k-2}\overline{\sigma}_{k-2}}^2 + \mathbf{g}_{k-2}\overline{\sigma}_{k-2} \qquad (2.40) \\
E\|\overline{\mathbf{h}}_{k-1}\|_{\overline{\sigma}_{k-1}}^2 &= E\|\overline{\mathbf{h}}_{k-2}\|_{\overline{F}_{k-1}\overline{\sigma}_{k-1}}^2 + \mathbf{g}_{k-1}\overline{\sigma}_{k-1} \\
&\vdots \\
E\|\overline{\mathbf{h}}_N\|_{\overline{\sigma}_N}^2 &= E\|\overline{\mathbf{h}}_{N-1}\|_{\overline{F}_N\overline{\sigma}_N}^2 + \mathbf{g}_N\overline{\sigma}_N \qquad (2.41)
\end{aligned}$$

These equations are solved via suitable choice of the free parameters $\{\overline{\sigma}_k\}$ for MSD and EMSE. By choosing $\overline{\sigma}_{k-2} = \overline{F}_{k-1}\overline{\sigma}_{k-1}$ we will obtain

$$E\|\overline{\mathbf{h}}_{k-1}\|^2_{(I-\Phi_{k,1})\overline{\sigma}_{k-1}} = \mathbf{a}_k\overline{\sigma}_{k-1} \tag{2.42}$$

where $\Phi_{k,l}$ and the row vector $\mathbf{a}_k$ are defined as

$$\Phi_{k,l} = \overline{F}_{k+l-1}\overline{F}_{k+l}\ldots\overline{F}_N\overline{F}_1\ldots\overline{F}_{k-1}, \quad l = 1, 2, \ldots, N \tag{2.43a}$$

$$\mathbf{a}_k = \mathbf{g}_k\Phi_{k,2} + \mathbf{g}_{k+1}\Phi_{k,3} + \ldots + \mathbf{g}_{k-2}\Phi_{k,N} + \mathbf{g}_{k-1} \tag{2.43b}$$

Now using the definitions of MSD, EMSE and MSE defined in (2.39) and selecting the weighting vector $\overline{\sigma}_{k-1}$ properly in (2.42) we arrive at following three expressions for the desired steady-state parameters.

$$\eta_k = \mathbf{a}_k(\mathbf{I} - \Phi_{k,1})^{-1}q \quad (\mathbf{MSD}) \tag{2.44}$$

$$\zeta_k = \mathbf{a}_k(\mathbf{I} - \Phi_{k,1})^{-1}\lambda_k \quad (\mathbf{EMSE}) \tag{2.45}$$

$$\xi_k = \zeta_k + \sigma^2_{v,k} \quad (\mathbf{MSE}) \tag{2.46}$$

$$\tag{2.47}$$

For small step sizes, $\overline{F}_k \approx \mathrm{I} - 2\mu_1\wedge_1$, i.e. $\overline{F}_k$ becomes a diagonal matrix. As a result, matrix $\Phi_{k,l} = \Phi = \overline{F}_1\overline{F}_2\ldots\overline{F}_N$ will also be diagonal and can be approximated as

$$\begin{aligned}\Phi &= (\mathrm{I} - 2\mu_1\wedge_1)(\mathrm{I} - 2\mu_2\wedge_2)\ldots(\mathrm{I} - 2\mu_N\wedge_N)\\ &\approx \mathrm{I} - (2\mu_1\wedge_1 + 2\mu_2\wedge_2 + \ldots + 2\mu_N\wedge_N)\end{aligned}$$

so that

$$\mathrm{I} - \Phi \approx 2\mu_1\wedge_1 + 2\mu_2\wedge_2 + \ldots + 2\mu_N\wedge_N$$

and $\mathbf{a}_k \approx \sum_{k=1}^N g_k$. From (2.45), we get

$$\eta_k \approx (\mu_1^2\sigma^2_{v,1}\lambda_1^T + \ldots + \mu_N^2\sigma^2_{v,N}\lambda_N^T)\,(2\mu_1\wedge_1 + \ldots + 2\mu_N\wedge_N)^{-1}q \tag{2.48}$$

In a similar way, for small step sizes, the EMSE can be approximated as

$$\zeta_k \approx (\mu_1^2\sigma^2_{v,1}\lambda_1^T + \ldots + \mu_N^2\sigma^2_{v,N}\lambda_N^T)\,(2\mu_1\wedge_1 + \ldots + 2\mu_N\wedge_N)^{-1}\lambda_k \tag{2.49}$$

Both MSE and EMSE go to zero asymptotically when the step size $\mu_l \to 0$, causing the MSE to achieve the background noise level $\sigma^2_{v,k}$ everywhere.

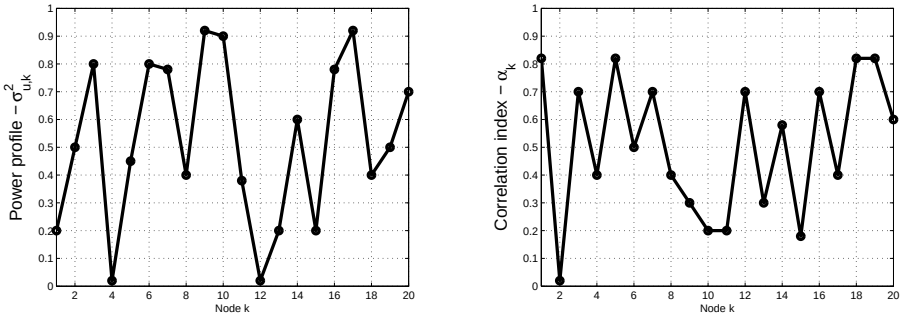
2.4 Results and Discussion

In order to evaluate the performance of the distributed incremental LMS algorithm, we provide the simulations comparing theoretical performance to simulation results. All simulations are carried out using regressors with shift structure. The desired data are generated according to the model given in (2.2) and the unknown vector $\mathbf{w}^\circ$ is set as $\mathbf{w}^\circ = [1, 1, \ldots, 1]^T/\sqrt{M}$.

The network consists of $N = 20$ nodes with each regressor of size (1×10) collecting data by observing a time-correlated sequence $u_k(i)$, generated as

$$u_k(i) = \alpha_k \cdot u_k(i-1) + \beta_k \cdot z_k(i), \qquad i > -\infty \tag{2.50}$$

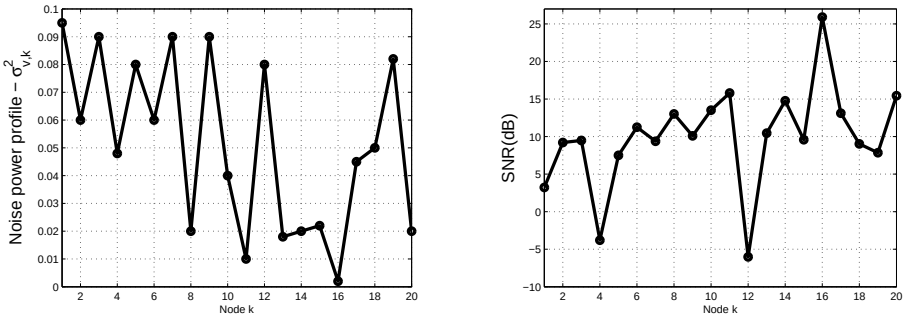
Here, $\alpha_k \in [0, 1)$ is the correlation index and $z_k(i)$ is a spatially independent white Gaussian process with unit variance and $\beta_k = \sqrt{\sigma^2_{u,k} \cdot (1 - \alpha^2_k)}$. The resulting regressors have Toeplitz covariance matrices with co-relation sequence $r_k(i) = \sigma^2_{u,k} \cdot (\alpha_k)^{|i|}, i = 0, 1, 2, \ldots, M-1$. The regressor power profile $\{\sigma^2_{u,k}\} \in (0, 1]$, the correlation indexes $\{\alpha^2_k\} \in [0, 1)$ and the Gaussian noise variances $\{\sigma^2_{v,k}\} \in (0, 0.1]$ were chosen at random and are depicted in Fig.2.1(a) and 2.1(b) and Fig.2.2(a) respectively. The corresponding signal-to-noise ratio (SNR) is plotted in Fig.2.2(b).



(a) (b)

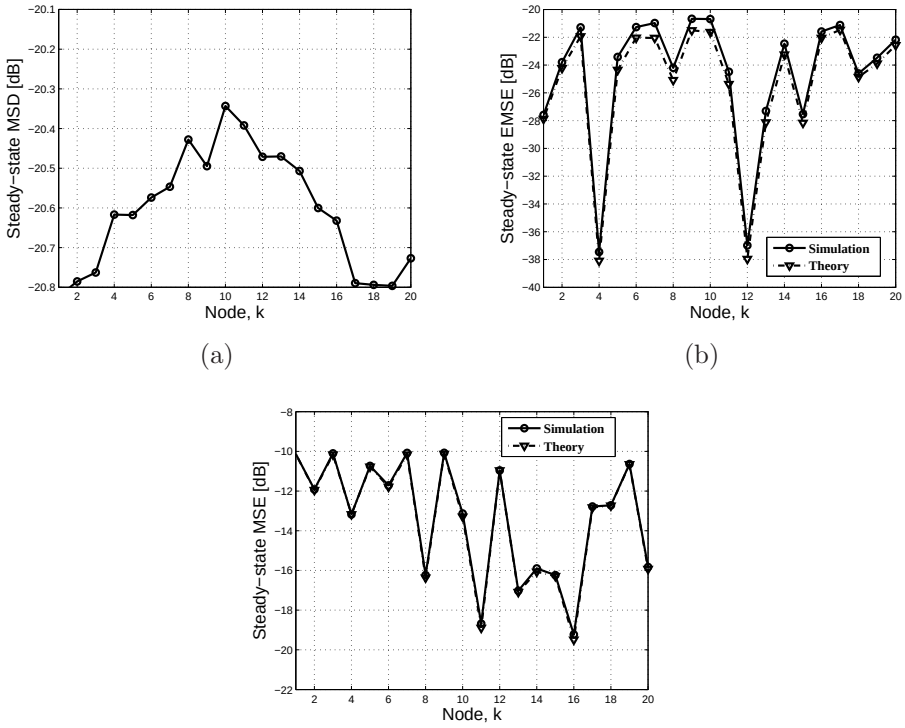
Figure 2.1: (a) Regressor power profile. (b) Correlation index per node

In order to generate the performance curves, 10 independent experiments were performed and averaged. The steady-state curves are generated by running the network



(a) (b)

Figure 2.2: (a) Noise power profile. (b) Signa-to-noise ration profile



(a) (b)

(c)

Figure 2.3: (a) MSD Vs nodes for $\mu = 0.03$. (b) EMSE Vs nodes for $\mu = 0.03$ (c) MSE Vs nodes for $\mu = 0.03$

learning process for 3000 iterations. The quantities MSD, EMSE and MSE are obtained by averaging the last 300 samples of the corresponding learning curves.

The results depict the steady-state quantities as a function of node k for particular value of the step size μ_k. These curves tell the designer how to adjust the step size at a certain node to compensate for a signal power increase in nearby nodes. It is observed

that there is a close match between theory and simulations.

A second kind of curve is provided that shows the behavior of the steady-state quantities as a function of the step size for a particular node. These curves evaluate the quality of theoretical model and the assumptions we have used to derive the theoretical model. A large deviations between theory and simulation are expected for bigger step sizes because when the step-size becomes large, the simplifying assumptions adopted in the analysis are no longer reasonable. The Figs. 2.3(a), 2.3(b) and 2.3(c) shows performance of the network as a function of the step-size.

A good step size design, together with the cooperative scheme, may take advantage of the spatial diversity provided by the adaptive network. A good level of performance equalization can be achieved throughout the network by proper tuning of the step size at each node. The nodes presenting poor performance, or high noise level should assign with small step-size to equalize the performance.

2.5 Conclusion

This chapter introduced a distributed incremental approach for estimation in wireless sensor network. The steady-state performance of incremental LMS algorithm was presented. The spatial energy-conservation arguments were used to study the steady-state performance of the network. The steady-state expressions for MSE, EMSE and MSD was derived and was found to be matching very well with simulation results.

Chapter 3

Distributed Diffusion Least Mean Square Algorithm

3.1 Introduction

A wireless sensor network consists of groups of sensors or nodes using wireless links to perform distributed sensing tasks by coordinating themselves. Distributed signal processing deals with the extraction of information from data collected at nodes that are distributed over a geographical area. Each node in a network collects noisy observations related to a certain parameter. The nodes would then interact with their neighbors in a certain manner according to the network topology either by incremental way [4] or by diffusion [3]. In a traditional centralized solution, the node in the network collects observations and conveys then to central processor where they would be fused and the vector of parameters estimated, then broadcast the result back to the individual node. This mode of operation requires a powerful central processor and more communication between nodes and the processor. In addition, a centralized solution may limit the ability of the nodes to adapt in real time.

Here a new type of cooperative algorithms is considered [3] that adopts diffusion protocol, in which nodes from the same neighborhood are allowed to communicate with each other. A network is more efficient if it requires less communication between nodes to estimate some vector of parameters [11],[1].

3.2 Diffusion LMS

We would like to estimate an $M \times 1$ unknown vector $\mathbf{w}^{\circ}$ from measurements collected at N nodes spread over a network (see Fig. 1). Each node k has access to time realizations $\{d_k(j), \mathbf{u}_{k,j}\}$ of zero mean random data $\{d_k, \mathbf{u}_k\}, k = 1, 2, 3, ...N$, where $d_k(j)$ is scalar measurement and $\mathbf{u}_{k,j}$ a $1 \times M$ regression row vector, both at time j is given as.

$$\mathbf{u}_{k,j} = [u_k(j), u_k(j-1), \dots u_k(j-M+1)]$$

We collect the regression and measurement data across all nodes into two global matrices where we drop the time index for compactness of notation.

$$\mathbf{U}_c = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N]^T \tag{3.1a}$$

$$\mathbf{d} = [d_1, d_2, \dots, d_N]^T \tag{3.1b}$$

Now we can define the autocorrelation matrix $\mathrm{R}_u = E[\mathbf{U}_c^* \mathbf{U}_c]$ and cross-correlation matrix $\mathrm{R}_{du} = E[\mathbf{U}_c^* \mathbf{d}]$ where E is the expectation operator. So we seek to minimize the equation

$$\min_{\mathbf{w}} E \parallel \mathbf{d} - \mathbf{U}_c \mathbf{w} \parallel^2 \tag{3.2}$$

The optimal solution $\mathbf{w}^\circ$ of (3.2) satisfies the orthogonality condition

$$E\mathbf{U}_c^*(\mathbf{d} - \mathbf{U}_c \mathbf{w}^\circ) = 0 \tag{3.3}$$

and is given as [12]

$$\mathbf{w}^\circ = \mathrm{R}_u^{-1} \mathrm{R}_{du} \tag{3.4}$$

For later reference, we also introduce the block diagonal matrix

$$\mathbf{U} = \mathrm{diag}\{\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_N\} \qquad (N \times NM) \tag{3.5}$$

and

$$\mathbf{Q} = [\mathrm{I}_M, \mathrm{I}_M, \ldots, \mathrm{I}_M]^T \qquad (NM \times M) \tag{3.6}$$

where I_M is the $M \times M$ identity matrix which is related U_c as

$$\mathbf{U}_c = \mathbf{U}\mathbf{Q} \tag{3.7}$$

3.2.1 Diffusion LMS algorithm

Our objective is to develop an adaptive distributed algorithm that allows cooperation among the nodes through limited local communications and gives the approximate solution $\mathbf{w}^\circ$ of (3.4). In addition it should deliver a good estimate of that vector at every node in the network. Here we develop a diffusion protocol where every node k in the network continuously combines estimates from its neighborhood. Specifically, at any given time $j-1$, we assume that node k has access to a set of unbiased estimates $\{\psi_k^{(j-1)}\}_{k \in \mathcal{N}_k}$ from its neighborhood $\mathcal{N}_k$, which is defined as the set of all nodes connected to it, including itself. The estimates $\{\psi_k^{(j-1)}\}_{k \in \mathcal{N}_k}$ are generally noisy versions of $\mathbf{w}^0$, say

$$\psi_k^{(j-1)} = \mathbf{w}^\circ - \tilde{\psi_k}^{(j-1)} \tag{3.8}$$

for some error $\psi_k^{(j-1)}$. These local estimates are fused at node k, yielding

$$\phi_k^{(j-1)} = f_k\left(\psi_l^{(j-1)}; l \in \mathcal{N}_{k,j-1}\right) \tag{3.9}$$

for some local combiner function f_k. Here we employ a linear combiner, and replace f_k by some weighted combination as

$$\phi_k^{(j-1)} = \sum_{l \in \mathcal{N}_{k,j-1}} c_{kl}\psi_l^{(j-1)} \tag{3.10}$$

for some combination coefficients $\{c_{kl} \geq 0\}$ to be determined from the network topology. We define the combining coefficients such that

$$\sum_l c_{kl} = 1, \qquad l \in \mathcal{N}_{k,j-1} \qquad \forall k \tag{3.11}$$

Once we have an aggregate estimate $\phi_k^{(j-1)}$ for $\mathbf{w}^\circ$, and in order to foster adaptivity, we subsequently fuse the resulting estimate $\phi_k^{(j-1)}$ into the local adaptive process, so that it can rapidly respond to changes in its neighborhood and update it to $\psi_k^{(j)}$. Analysis and simulation will show that this scheme leads to a robust distributed adaptive system that achieves smaller error levels in steady-state than its noncooperative counterpart [3].

The proposed diffusion strategy can therefore be described in general form as follows:

$$\phi_k^{(j-1)} = f_k\left(\psi_l^{(j-1)}; l \in \mathcal{N}_{k,j-1}\right) \tag{3.12}$$

$$\psi_k^{(j)} = \phi_k^{(j-1)} + \mu_k \mathbf{u}_{k,j}^*\left(d_k(j) - \mathbf{u}_{k,j}\phi_k^{(j-1)}\right) \tag{3.13}$$

for local step sizes μ_k. The combiners may be nonlinear, or even time-variant, to reflect the changing topologies. This can be implemented by assuming the neighborhood $\mathcal{N}_k$ to be time variant. The resulting adaptive network is a peer-to-peer estimation framework and robust to node and link failures.

Here, we exploit a linear combiner model with LMS-type adaptive algorithm. The algorithm is given below :

$$\phi_k^{(j-1)} = \sum_{l \in \mathcal{N}_{k,j-1}} c_{kl}\psi_l^{(j-1)}, \qquad \phi_k^{(-1)} = 0 \tag{3.14a}$$

$$\psi_k^{(i)} = \phi_k^{(j-1)} + \mu_k \mathbf{u}_{k,j}^*\left(d_k(j) - \mathbf{u}_{k,j}\phi_k^{(j-1)}\right) \tag{3.14b}$$

for a set of *local* combiners c_{kl} satisfying (3.11).

3.3 Network Global Model

The algorithm (3.14) embeds the combined effect of several interconnected adaptive filter updates. We introduce the global quantities as

$$\psi^j = [\psi_1^{(j)}, \ldots, \psi_N^{(j)}]^T, \qquad \phi^j = [\phi_1^{(j)}, \ldots, \phi_N^{(j)}]^T$$

$$\mathbf{U}_j = \text{diag}\{\mathbf{u}_{1,j}, \ldots, \mathbf{u}_{N,j}\}, \qquad \mathbf{d}_j = [\mathbf{d}_{1,j}, \ldots, \mathbf{d}_{N,j}]^T$$

Let

$$D = \text{diag}\{\mu_1 \mathrm{I}_M, \mu_2 \mathrm{I}_M, \ldots, \mu_N \mathrm{I}_M\} \qquad (NM \times NM) \tag{3.15}$$

be a diagonal matrix collecting the local step sizes. The measurements are assumed to obey a traditional model of the form [6],[5], [10]

$$\mathbf{d}_k(j) = \mathbf{u}_{k,j}\mathbf{w}^\circ + \mathbf{v}_k(j) \tag{3.16}$$

where $\mathbf{v}_{k,j}$ is background noise, assumed independent over time and space and with variance $\sigma_{v,k}^2$. Linear models of the form (3.16) are able to capture or approximate many input output relations for estimation purposes. We can write the global linear model as

$$\mathbf{d}_j = \mathbf{U}_j w^{(\circ)} + \mathbf{v}_j \tag{3.17}$$

where $\mathbf{w}^{(\circ)} = Q\mathbf{w}^\circ$ and

$$\mathbf{v}_j = [\mathbf{v}_{1,j}, \ldots, \mathbf{v}_{N,j}]^T \qquad (N \times 1)$$

With these relations, the algorithm (3.14) can be written in global form:

$$\begin{aligned} \phi^{j-1} &= G\psi^{j-1} \\ \psi^j &= \phi^{j-1} + D\mathbf{U}_j^*(\mathbf{d}_j - \mathbf{U}_j\phi^{j-1}) \end{aligned} \tag{3.18}$$

or in a more compact form:

$$\psi^j = G\psi^{j-1} + D\mathbf{U}_j^*(\mathbf{d}_j - \mathbf{U}_j G\psi^{j-1}) \tag{3.19}$$

where $G = C \otimes I_M$ is the $NM \times NM$ transition matrix and C is the $N \times N$ diffusion combination matrix with entries $[c_{kl}]$.

Combiner Coefficient

We should define C such that $C_{qN} = col\{1, 1, \ldots, 1\}$. Possible choices for combiner C are the Metropolis, the Laplacian and the nearest neighbor rules [3]. The metropolis rule is defined as follows.

Let n_k and n_l denote the degree for nodes k and l, i.e., $n_k =\mid \mathcal{N}_k \mid$, and choose

$$c_{kl} = \begin{cases} \frac{1}{\max(n_k, n_l)}, & \text{if } k \neq l \text{ are linked} \\ 0, & \text{for } k \text{ and } l \text{ not linked} \\ 1 - \sum_{l \in \mathcal{N}_k / k} c_{kl}, & \text{for } k = l \end{cases} \tag{3.20}$$

The Laplacian rule is given by

$$C = \mathrm{I}_N - \mathcal{K}\mathcal{L} \tag{3.21}$$

where $\mathcal{L} = \mathcal{D} - A_d$, with $\mathcal{D} = \mathrm{diag}\{n_1, n_2, \ldots, n_N\}$, $\mathcal{K} = \frac{1}{n_{\max}}$ and A_d is the $N \times N$ network adjacent matrix formed as

$$[A_d]_{kl} = \begin{cases} 1, & \text{if } k \text{ and } l \text{ are linked} \\ 0, & \text{otherwise} \end{cases} \tag{3.22}$$

For the nearest neighbour rule, the combiner matrix C is defined as

$$c_{kl} = \begin{cases} \frac{1}{|\mathcal{N}_k|}, & l \in \mathcal{N}_k \\ 0, & \text{otherwise} \end{cases} \tag{3.23}$$

3.4 Performance Analysis

The performance analysis in an interconnected network is a challenging job due to the following reasons:

1. each node k is influenced by the local data with local statistics $\{R_{yx,k}, R_{x,k}\}$
2. each node k is influenced by its neighborhood nodes through local diffusion
3. each node is influenced by the local noise with variance $\sigma^2_{v,k}$

The energy based approach is extended to the space dimension because the distributed adaptive algorithm (3.14) involves both the time variable j(block number) and space variable k. We will define the common terms MSD(Mean Square Deviation), MSE(Mean Square Error) and EMSE(Excess Mean Square Error) for local and also for global network.

3.4.1 Mean Transient Analysis

Introducing the global weight error vector

$$\tilde{\psi}^j = \mathbf{w}^{(\circ)} - \psi^j \tag{3.24}$$

Since, $G\mathbf{w}^{(\circ)} = \mathbf{w}^{(\circ)}$, using global data model (3.17) and subtracting $\mathbf{w}^{\circ}$ from the left hand side and $G\mathbf{w}^{(\circ)}$ from the right side of (3.19), we get

$$\tilde{\psi}^j = G\tilde{\psi}^{j-1} - DU_j^* \left(U_j \mathbf{w}^{(\circ)} + v_j - U_j G\tilde{\psi}^{j-1} \right)$$

We can write this in compact form as

$$\tilde{\psi}^j = \left(\mathrm{I}_{NM} - DU_j^* U_j \right) G\tilde{\psi}^{j-1} - DU_j^* \mathbf{v}_j \tag{3.25}$$

Assuming temporal and spatial independence of the regression data $\{\mathbf{u}_{k,j}\}$ and taking the expectations of both sides of the above equation (3.25) leads to

$$E[\tilde{\psi}^j] = (\mathrm{I}_{NM} - D\mathrm{R}_u)\, GE[\tilde{\psi}^{j-1}] \tag{3.26}$$

where $\mathrm{R}_u = \mathrm{diag}\{\mathrm{R}_{u,1}, \mathrm{R}_{u,2}, \ldots, \mathrm{R}_{u,N}\}$ is block diagonal and $\mathrm{R}_{u,k} = E[\mathbf{u}_{k,j}^* \mathbf{u}_{k,j}]$. In the absence of cooperation(i.e., when the nodes evolve independently of each other and therefore $G = I_{NM}$), the mean error vector would evolve according to

$$E[\tilde{\psi}^j] = (\mathrm{I}_{NM} - D\mathrm{R}_u)\, E[\tilde{\psi}^{j-1}] \tag{3.27}$$

3.4.2 Mean-Square Transient Analysis

We proceed to perform transient analysis of the adaptive network and characterize the evolution of its learning curves. Here we will derive the expressions for mean-square-deviation(MSD) and excess-mean-square-deviation(EMSE).

We can define the local output estimation error at node k as

$$\mathbf{e}_k^j = \mathbf{d}_k^j - \mathbf{u}_{k,j}\phi_k^{(j-1)} \tag{3.28}$$

Now the global error vector can be written by collecting the errors across the network as:

$$\mathbf{e}_j = [\mathbf{e}_{1,j}, \mathbf{e}_{2,j}, \ldots, \mathbf{e}_{N,j}]^T \qquad (N \times 1)$$

so that we can write

$$\mathbf{e}_j = \mathbf{d}_j - \mathbf{U}_j G\Psi^{(j-1)} = \mathbf{U}_j G\tilde{\Psi}^{(j-1)} + \mathbf{v}_j = \mathbf{e}_{a,j}^G + \mathbf{v}_j \tag{3.29}$$

where

$$\mathbf{e}_{a,j}^G = \mathbf{U}_j G\tilde{\Psi}^{(j-1)} \tag{3.30}$$

Introducing global *a priori* and *a posteriori* weighted estimation errors:

$$\mathbf{e}_{a,j}^{D\Sigma G} = \mathbf{U}_j D\Sigma G\tilde{\psi}^{(j-1)} \qquad \text{and} \qquad \mathbf{e}_{p,j}^{D\Sigma} = \mathbf{U}_j D\Sigma\tilde{\psi}^j \tag{3.31}$$

for some arbitrary $NM \times NM$ matrix $\Sigma \geq 0$. The freedom in selecting Σ enable us to characterize MSD and EMSE performance of the network. Now we redefine the global weight error vector as

$$\begin{aligned}\tilde{\psi}^j = \mathbf{w}^\circ - \Psi^j &= G\mathbf{w}^{(\circ)} - G\Psi^{j-1} - D\mathbf{U}_j^*(\mathbf{d}_j - \mathbf{U}_j G\Psi^{j-1}) \\ &= G\tilde{\Psi}^{j-1} - D\mathbf{U}_j^*\mathbf{e}_j\end{aligned} \tag{3.32}$$

Multiplying $\mathbf{U}_j D\Sigma$ on both sides of (3.32)

$$\mathbf{e}_{p,j}^{D\Sigma} = \mathbf{e}_{a,j}^{D\Sigma G} - \| \mathbf{U}_j \|_{D\Sigma D}^2 \mathbf{e}_j \tag{3.33}$$

therefore the global error can be written as:

$$\mathbf{e}_j = \frac{\mathbf{e}_{a,j}^{D\Sigma G} - \mathbf{e}_{p,j}^{D\Sigma}}{\| \mathbf{U}_j \|_{D\Sigma D}^2} \tag{3.34}$$

using the above equation (3.34) in (3.32), we get

$$\tilde{\Psi}^j = G\tilde{\Psi}^{j-1} - D\mathbf{U}_j^* \frac{\mathbf{e}_{a,j}^{D\Sigma G} - \mathbf{e}_{p,j}^{D\Sigma}}{\| \mathbf{U}_j \|_{D\Sigma D}^2}$$

$$\tilde{\Psi}^j + \frac{D\mathbf{U}_j^* \mathbf{e}_{a,j}^{D\Sigma}}{\| \mathbf{U}_j \|_{D\Sigma D}^2} = G\tilde{\Psi}^{j-1} + \frac{D\mathbf{U}_j^* \mathbf{e}_{p,j}^{D\Sigma G}}{\| \mathbf{U}_j \|_{D\Sigma D}^2} \tag{3.35}$$

Equating the weighted norm of both sides of (3.35), we find that the cross terms cancel out, and we end up with the energy terms,i.e.,

$$\| \tilde{\Psi}^j \|_{\Sigma}^2 + \frac{\mathbf{e}_{a,j}^{D\Sigma 2}}{\| \mathbf{U}_j \|_{D\Sigma D}^2} = \| \tilde{\Psi}^{j-1} \|_{G^*\Sigma G}^2 + \frac{\mathbf{e}_{p,j}^{D\Sigma 2}}{\| \mathbf{U}_j \|_{D\Sigma D}^2} \tag{3.36}$$

Equation (3.36) is a $space - time$ version of the weighted energy conservation relation. This is an exact relation that shows how energies of several error variables are related to each other in space and time.

Variance Relation

Replacing $\mathbf{e}_{p,j}^{D\Sigma}$ by its definition in (3.33), we get

$$\| \tilde{\psi}^j \|_{\Sigma}^2 = \| \tilde{\psi}^{j-1} \|_{G^*\Sigma G}^2 - \mathbf{e}_j^* \mathbf{e}_{a,j}^{D\Sigma G} - (\mathbf{e}_{a,j}^{D\Sigma G})^* \mathbf{e}_j + \| \mathbf{U}_j \|_{D\Sigma D}^2 \mathbf{e}_j^* \mathbf{e}_j \tag{3.37}$$

Using error relation (3.29) and taking expectation on both sides and then using definitions of weighted errors (3.31), we get

$$\begin{aligned} E\| \tilde{\psi}^j \|_{\Sigma}^2 = E\| \tilde{\psi}^{j-1} \|_{G^*\Sigma G}^2 - E\| \tilde{\psi}^{j-1} \|_{G^*\Sigma D\mathbf{U}_j^*\mathbf{U}_j G}^2 - E\| \tilde{\psi}^{j-1} \|_{G^*\mathbf{U}_j^*\mathbf{U}_j D\Sigma G}^2 \\ + E\| \tilde{\psi}^{j-1} \|_{G^*\mathbf{U}_j^*\mathbf{U}_j D\Sigma D\mathbf{U}_j^*\mathbf{U}_j G}^2 + E\mathbf{v}_j^*\mathbf{U}_j D\Sigma D\mathbf{U}_j^*\mathbf{v}_j \end{aligned} \tag{3.38}$$

Now, using the relation $\| x \|_A^2 + \| x \|_B^2 = \| x \|_{A+B}^2$, the previous equation can be written more compactly as

$$E\| \tilde{\psi}^j \|_{\Sigma}^2 = E\left(\| \tilde{\psi}^{j-1} \|_{\mathbf{\Sigma}'}^2 \right) + E\mathbf{v}_j^*\mathbf{U}_j D\Sigma D\mathbf{U}_j^*\mathbf{v}_j \tag{3.39}$$

where

$$\mathbf{\Sigma}' = G^*\Sigma G - G^*\Sigma D\mathbf{U}_j^*\mathbf{U}_j G - G^*\mathbf{U}_j^*\mathbf{U}_j D\Sigma G - G^*\mathbf{U}_j^*\mathbf{U}_j D\Sigma D\mathbf{U}_j^*\mathbf{U}_j G \tag{3.40}$$

Note that no assumptions are needed to arrive at (3.39)-(3.40). However, the weighting matrix $\mathbf{\Sigma}'$ is data dependent. This makes the analysis very challenging, so that some assumptions need to be introduced for mathematical tractability. So we proceed by assuming temporal and spatial independence of regression data, so that $\mathbf{U}_j$ is independent of $\tilde{\Psi}^{j-1}$, as is common in the analysis of traditional adaptive schemes[6],[5], [10]. In this way the random weighting matrix $\mathbf{\Sigma}'$ can be replaced by its mean value (a deterministic matrix $\Sigma' = E\mathbf{\Sigma}'$). So

$$E\left(\| \tilde{\psi}^{j-1} \|^2_{\mathbf{\Sigma}'}\right) = E\| \tilde{\psi}^{j-1} \|^2_{E\mathbf{\Sigma}'} \tag{3.41}$$

which reduces the equations (3.39)-(3.40) to the following variance relation

$$E\| \tilde{\Psi}^{j} \|^2_{\Sigma} = E\| \tilde{\psi}^{j-1} \|^2_{\Sigma'} + E\mathbf{v}_j^* \mathbf{U}_j D\Sigma D\mathbf{U}_j^* \mathbf{v}_j \tag{3.42}$$

where $\Sigma' = E\mathbf{\Sigma}'$ is given by

$$\Sigma' = G^*\Sigma G - G^*\Sigma DE(\mathbf{U}_j^*\mathbf{U}_j)G - G^*E(\mathbf{U}_j^*\mathbf{U}_j)D\Sigma G + G^*E\mathbf{U}_j^*\mathbf{U}_j D\Sigma D\mathbf{U}_j^*\mathbf{U}_j G \tag{3.43}$$

Gaussian Regressor

Assume that the regressors arise from a circular Gaussian distribution. Now introduce the eigendecomposition $\mathbf{R}_u = T \wedge T^*$, where $\wedge$ is a diagonal matrix with eigenvalues of $\mathbf{R}_u$. Then define the transformed quantities :

$$\overline{\Psi}^j = T^*\tilde{\Psi}^j \qquad \overline{\mathbf{U}}_j = \mathbf{U}_j T \qquad \overline{G} = T^*GT$$

$$\overline{\Sigma} = T^*\Sigma T \qquad \overline{\Sigma}' = T^*\Sigma' T \qquad \overline{D} = T^*DT = D$$

where $\overline{D} = D$ follows from (3.15). Now under the change of variables, the variance relation (3.42) and (3.43) can be written as

$$E\| \overline{\Psi}^{j} \|^2_{\overline{\Sigma}} = E\| \overline{\psi}^{j-1} \|^2_{\overline{\Sigma}'} + E\mathbf{v}_j^* \overline{\mathbf{U}}_j D\overline{\Sigma} D\overline{\mathbf{U}}_j^* \mathbf{v}_j \tag{3.44}$$

$$\overline{\Sigma}' = \overline{G}^*\overline{\Sigma}\overline{G} - \overline{G}^*\overline{\Sigma}DE(\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j)\overline{G} - \overline{G}^*E(\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j)D\overline{\Sigma}\overline{G} + \overline{G}^*E(\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma} D\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j)\overline{G} \tag{3.45}$$

Let Σ be an $NM \times NM$ block matrix

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \dots & \Sigma_{1l} & \dots & \Sigma_{1N} \\ \Sigma_{21} & \Sigma_{22} & \dots & \Sigma_{2l} & \dots & \Sigma_{2N} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \Sigma_{N1} & \Sigma_{N2} & \dots & \Sigma_{Nl} & \dots & \Sigma_{NN} \end{bmatrix} \tag{3.46}$$

where each block Σ_{kl} is $M \times M$. First, the block columns

$$\Sigma_l = [\Sigma_{1l}, \Sigma_{2l}, \dots, \Sigma_{Nl}]^T \qquad l = 1, \dots, N$$

are stacked on top of each other, yielding the $N^2M \times M$ matrix

$$\Sigma^c = \begin{bmatrix} \Sigma_1 \\ \Sigma_2 \\ \vdots \\ \Sigma_N \end{bmatrix} \tag{3.47}$$

Subsequently we move along Σ^c and vectorize each individual block Σ_{kl} via the standard vec$\{\}$ operator, so that for each stacked block column Σ_l, we get

$$\sigma_l = [\sigma_{1l}, \sigma_{2l}, \dots, \sigma_{Nl}]^T \qquad \text{with} \qquad \sigma_{kl} = \text{vec}\{\Sigma_{kl}\} \tag{3.48}$$

The final vectorized matrix is obtained from

$$\sigma = [\sigma_1, \sigma_2, \dots, \sigma_N]^T \tag{3.49}$$

We thus write $\sigma = \text{bvec}\{\Sigma\}$ to denote the conversion of Σ into a single column.We also write $\Sigma = \text{bvec}\{\sigma\}$ to recover the original block matrix form of the column vector σ. The *block Kronecker product* of two block matrices A and B is denoted by $A \odot B$. Its kl-block is defined as

$$[A \odot B]_{kl} = \begin{bmatrix} A_{kl} \otimes B_{11} & \dots & A_{kl} \otimes B_{1N} \\ \vdots & \ddots & \vdots \\ A_{kl} \otimes B_{N1} & \dots & A_{kl} \otimes B_{NN} \end{bmatrix} \tag{3.50}$$

for $k, l = 1, \ldots, N$. The block vector operator (3.49) and the block Kronecker product (3.50) are related via

$$\text{bvec}\{A\Sigma B\} = (B \odot A^T)\sigma \tag{3.51}$$

The first moment in (3.45) is trivial and given by $E\overline{\mathbf{U}}_j^T\overline{\mathbf{U}}_j = \wedge$. By using (3.51) and after vectorization, the second and third terms on the RHS of 3.45 are given by

$$\begin{aligned}\text{bvec}\{\overline{G}^*\overline{\Sigma}D\wedge\overline{G}\} &= \left(\overline{G}\odot\overline{G}^{*T}\right)\text{bvec}\{I_{NM}\overline{\Sigma}D\wedge\} \\ &= \left(\overline{G}\odot\overline{G}^{*T}\right)(\wedge D\odot I_{NM})\overline{\sigma}\end{aligned} \tag{3.52}$$

and

$$\text{bvec}\{\overline{G}^*\overline{\Sigma}D\overline{\Sigma}\overline{G}\} = \left(\overline{G}\odot\overline{G}^{*T}\right)(I_{NM}\odot\wedge D)\overline{\sigma} \tag{3.53}$$

The moment in (3.44) can be verified to be

$$E\mathbf{v}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\mathbf{v}_j = \text{Tr}\{\wedge_v E\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\} \tag{3.54}$$

where $\wedge_v > 0$ is a diagonal matrix given by

$$\wedge_v = \text{diag}\{\sigma_{v,1}^2, \sigma_{v,2}^2, \cdots, \sigma_{v,N}^2\}$$

The kl entry of $E\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*$ is given by

$$[E\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*]_{kl} = \begin{cases} 0, & \text{for } k \neq l \\ \mu_k^2\text{Tr}(\wedge_k\overline{\Sigma}_{kk}) = \mu_k^2\lambda_k^T\overline{\sigma}_{kk}, & \text{for } k = l \end{cases} \tag{3.55}$$

where $\lambda_k = \text{vec}\{\wedge_k\}$ and $\overline{\sigma}_{kl} = \text{vec}\{\overline{\Sigma}_{kl}\}$, so that (3.54) gives

$$E\mathbf{v}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\mathbf{v}_j = b^T\overline{\sigma} \tag{3.56}$$

with $b = \text{bvec}\{R_v D^2\wedge\}$ and $R_v = \wedge_v \odot I_M$ and $\overline{\sigma} = \text{bvec}\{\overline{\Sigma}\}$ The fourth-order moment in (3.45) is challenging. To begin with

$$\text{bvec}\{\overline{G}^* E\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j\overline{G}\} = (\overline{G}\odot\overline{G}^{*T}).\text{bvec}\{E\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j\} \tag{3.57}$$

Now both $\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j$ and D are block diagonal, so that

$$E\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j = DE(\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j\overline{\Sigma}\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j)D \tag{3.58}$$

which gives

$$\text{bvec}\{\overline{G}^* E\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j D\overline{\Sigma}D\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j\overline{G}\} = (\overline{G}\odot\overline{G}^{*T})(D\odot D)\text{bvec}\{A\} \tag{3.59}$$

where

$$A = E\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j\overline{\Sigma}\overline{\mathbf{U}}_j^*\overline{\mathbf{U}}_j \tag{3.60}$$

The $M\times M$ kl-block of A is given by

$$A_{kl} = E\overline{\mathbf{u}}_{k,j}^*\overline{\mathbf{u}}_{k,j}\overline{\Sigma}_{k,l}E\overline{\mathbf{u}}_{l,j}^*\overline{\mathbf{u}}_{l,j} \quad = \begin{cases} \wedge_k\text{Tr}(\wedge_k\overline{\Sigma}_{kk}) + \gamma\wedge_k\overline{\Sigma}_{kk}\wedge_k, & \text{for } k\neq l \\ \wedge_k\overline{\Sigma}_{kl}\wedge_l, & \text{for } k = l \end{cases} \tag{3.61}$$

where $\gamma = 1$ for complex and $\gamma = 2$ for real data. Now express A as

$$A = [A_1, A_2, \ldots, A_l, \ldots, A_N] \tag{3.62}$$

where A_l is the l^{th} block column of A:

$$A_l = [A_{1l}, A_{2l}, \ldots, A_{kl}, \ldots, A_{Nl}]^T \tag{3.63}$$

It follows that

$$a = \text{bvec}\{A\} = [a_1, a_2, \ldots, a_l, \ldots, a_N]^T \tag{3.64}$$

where

$$a_l = [a_{1l}, a_{2l}, \ldots, a_{kl}, \ldots, a_{Nl}]^T \tag{3.65}$$

and

$$a_{kl} = \text{vec}\{A_{kl}\} = \begin{cases} (\lambda_k\lambda_k^T + \gamma\wedge_k\otimes\wedge_k)\overline{\sigma}_{kk}, & \text{for } k = l \\ (\wedge_k\otimes\wedge_l)\overline{\sigma}_{kl}, & \text{for } k\neq l \end{cases} \tag{3.66}$$

Hence

$$\begin{aligned} a_l &= [(\wedge_1 \otimes \wedge_l)\overline{\sigma}_{1l}, \ldots, (\lambda_l\lambda_l^T + \gamma \wedge_l \otimes \wedge_l)\overline{\sigma}_{1k}, \ldots, (\wedge_N \otimes \wedge_l)\overline{\sigma}_{Nl}]^T \\ &= \mathcal{A}_l\overline{\sigma}_l \end{aligned} \tag{3.67}$$

where

$$\mathcal{A}_l = \text{diag}\{\wedge_1 \otimes \wedge_l, \ldots, \lambda_l\lambda_l^T + \gamma \wedge_l \otimes \wedge_l, \ldots, \wedge_N \otimes \wedge_l\} \tag{3.68}$$

and $\overline{\sigma}_l = [\overline{\sigma}_{1l}, \overline{\sigma}_{2l}, \ldots, \overline{\sigma}_{Nl}]^T$. We thus find that

$$\text{bvec}\{A\} = [\mathcal{A}_1\overline{\sigma}_1, \mathcal{A}_2\overline{\sigma}_2, \ldots, \mathcal{A}_N\overline{\sigma}_N]^T = \mathcal{A}\overline{\sigma} \tag{3.69}$$

where

$$\mathcal{A} = \text{diag}\{\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_N\} \qquad \text{and} \qquad \overline{\sigma} = \text{bvec}\{\overline{\sigma}\}$$

In summary, grouping the results and substituting into (3.44),

$$E\| \overline{\Psi}^j \|_{\overline{\sigma}}^2 = E\| \overline{\psi}^{j-1} \|_{\overline{F}\overline{\sigma}}^2 + b^T\overline{\sigma} \tag{3.70}$$

$$\overline{F} = \left(\overline{G} \odot \overline{G}^{*T}\right)\left[I_{N^2M^2} - (I_{NM} \odot \wedge D) - (\wedge D \odot I_{NM}) + (D \odot D)\,\mathcal{A}\right]\overline{\sigma} \tag{3.71}$$

The compact notation $\| x \|_{\sigma}^2$ is used to refer to $\| x \|_{\Sigma}^2$, with the weighting matrix Σ replaced by its vector representation $\sigma = \text{bvec}\Sigma$.

Learning Behavior

Iterating (3.70) and (3.71), we get

$$\begin{aligned} E\| \overline{\Psi}^j \|_{\overline{\sigma}}^2 &= E\| \overline{\psi}^{j-1} \|_{\overline{F}\overline{\sigma}}^2 + b^T\overline{\sigma} \\ E\| \overline{\Psi}^{j-1} \|_{\overline{F}\overline{\sigma}}^2 &= E\| \overline{\psi}^{j-2} \|_{\overline{F}^2\overline{\sigma}}^2 + b^T\overline{F}\overline{\sigma} \\ E\| \overline{\Psi}^{j-2} \|_{\overline{F}^2\overline{\sigma}}^2 &= E\| \overline{\psi}^{j-3} \|_{\overline{F}^3\overline{\sigma}}^2 + b^T\overline{F}^2\overline{\sigma} \\ &\vdots \\ E\| \overline{\Psi}^0 \|_{\overline{F}^i\overline{\sigma}}^2 &= \| \overline{w}^{(0)} \|_{\overline{F}^{i+1}\overline{\sigma}}^2 + b^T\overline{F}^i\overline{\sigma} \end{aligned} \tag{3.72}$$

where $\overline{w}^{(0)} = T^* w^{(0)}$ and the last equality 3.72) follows from the fact that the local adaptive filters are initialized with zeros. Relations 3.72) lead to the result

$$E\| \overline{\Psi}^j \|^2_{\overline{\sigma}} = E\| \overline{w}^{(0)} \|^2_{\overline{F}^{j+1}\overline{\sigma}} + b^T \left(\sum_{k=0}^{j} \overline{F}^k \right) \overline{\sigma} \tag{3.73}$$

which in turn motivates the following useful recursions:

$$E\| \overline{\Psi}^j \|^2_{\overline{\sigma}} = E\| \overline{\Psi}^{j-1} \|^2_{\overline{\sigma}} + b^T \overline{F}^j \overline{\sigma} - \| \overline{w}^{(0)} \|^2_{\overline{F}^j (I-\overline{F})\overline{\sigma}} \tag{3.74}$$

Defining the global mean-square deviation as the average of the global quantity $E\| \overline{\Psi}^j \|^2$, i.e. $\eta(j) = \frac{1}{N} E\| \overline{\Psi}^j \|^2$. Therefore, choosing $\overline{\sigma} = \frac{1}{N}\text{bvec}\{I_{NM}\} = q_\eta$ in (3.74) leads to

$$\eta(j) = \eta(j-1) + b^T \overline{F}^j q_\eta - \| \overline{w}^{(0)} \|^2_{\overline{F}^j (I-\overline{F}) q_\eta} \tag{3.75}$$

with initial condition $\eta(-1) = \| \overline{w}^{(0)} \|^2$. In a similar manner choosing $\overline{\sigma} = \frac{1}{N}\text{bvec}\{\wedge\} = \lambda_\zeta$ leads to the global learning curve for the excess mean-square error $\zeta(j) = \frac{1}{N} E\| \overline{\Psi}^j \|_\wedge = \frac{1}{N} E\| e_{a,j} \|^2$, where $e_{a,j} = U_j \tilde{\Psi}^{j-1}$, so that

$$\zeta(j) = \zeta(j-1) + b^T \overline{F}^j \lambda_\zeta - \| \overline{w}^{(0)} \|^2_{\overline{F}^j (I-\overline{F})\lambda_\zeta} \tag{3.76}$$

with initial condition $\zeta(-1) = \| \overline{w}^{(0)} \|^2_\wedge$

Local Node performance

To extract the local quantities from the global expressions, the following spatial filtering matrices are defined as:

$$S_{q,k} = \text{diag}\{\mathbf{0}_{(k-1)M}, I_M, \mathbf{0}_{(N-k)M}\} \quad (\textbf{MSD}) \tag{3.77a}$$

$$S_{\lambda,k} = \text{diag}\{\mathbf{0}_{(k-1)M}, I_M, \mathbf{0}_{(N-k)M}\} \quad (\textbf{EMSE}) \tag{3.77b}$$

where $\wedge_k$ is the diagonal matrix with the eigen values corresponding to node k and $\mathbf{0}_L$ is a block of $L \times L$ zeros. Define the vectors

$$s_{q,k} = \text{bvec}\{S_{q,k}\} \qquad \text{and} \qquad s_{\lambda,k} = \text{bvec}\{S_{\lambda,k}\} \tag{3.78}$$

Selecting $\overline{\sigma}$ as the filtering vectors (3.78) in the global learning recursion (3.74) yields the local MSD at node k

$$\overline{f}_{q,j} = \overline{F}^{j}(I_{N^2M^2} - \overline{F})s_{q,k} \tag{3.79}$$

$$\eta_k(j) = \eta_k(j-1) + b^T\overline{F}^{j}s_{\lambda,k} - \| \overline{w}^{(0)} \|^2_{\overline{F}^{j}(I-\overline{F})\mathrm{bvec}\{\overline{f}_{q,j}\}} \tag{3.80}$$

with initial condition $\eta_k(-1) = \| \overline{w}^{(0)} \|^2_{S_{q,k}}$. Similarly, for the EMSE at node k is

$$\overline{f}_{\lambda,j} = \overline{F}^{j}(I_{N^2M^2} - \overline{F})s_{\lambda,k} \tag{3.81}$$

$$\zeta(j) = \zeta(j-1) + b^T\overline{F}^{j}s_{\lambda,k} - \| \overline{w}^{(0)} \|^2_{\overline{F}^{j}(I-\overline{F})\mathrm{bvec}\{\overline{f}_{\lambda,j}\}} \tag{3.82}$$

with initial condition $\zeta_k(-1) = \| \overline{w}^{(0)} \|^2_{S_{q,k}}$.

3.4.3 Steady State Analysis

Global Network Performance

The global seady-state quantities MSD and EMSE are defined as

$$\eta = \frac{1}{N}E\| \overline{\Psi}^{j-1} \|^2 \quad (\mathbf{MSD}) \tag{3.83a}$$

$$\zeta = \frac{1}{N}E\| \overline{\Psi}^{j-1} \|_{\wedge} \quad (\mathbf{EMSE}) \tag{3.83b}$$

as $j \to \infty$. In steady state, (3.70) leads to

$$E\| \overline{\Psi}^{\infty} \|^2_{(I-\overline{F})\overline{\sigma}} = b^T\overline{\sigma} \tag{3.84}$$

Considering two possibilities for $\overline{\sigma}$ defined by

$$(I - \overline{F})\overline{\sigma}_{\eta} = q \quad \text{and} \quad (I - \overline{F})\overline{\sigma}_{\zeta} = \lambda \tag{3.85}$$

They lead to

$$\eta = \frac{1}{N} b^T (I - \overline{F}) q \quad (\textbf{MSD}) \tag{3.86a}$$

$$\zeta = \frac{1}{N} b^T (I - \overline{F}) \lambda \quad (\textbf{EMSE}) \tag{3.86b}$$

Local Node Performance

The local mean-square performance of node k is defined as

$$\eta = E\| \overline{\Psi}_k^{\infty} \|^2 \quad \text{and} \quad \zeta = E\| \overline{\Psi}_k^{\infty} \|_{\lambda_k}^2 \tag{3.87}$$

Resorting to the filtering matrices (3.70) and rewriting the above local quantities in terms of the global quantities and the filtering matrices:

$$\eta = E\| \overline{\Psi}^{\infty} \|_{S_{q,k}}^2$$
$$\zeta = E\| \overline{\Psi}^{\infty} \|_{S_{\lambda,k}}^2$$

Selecting $\overline{\sigma}$ in (3.84) as the solutions to the linear systems of equations

$$(I - \overline{F})\overline{\sigma}_\eta = \text{bvec}\{S_{q,k}\} \quad (\textbf{MSD}) \tag{3.88a}$$

$$(I - \overline{F})\overline{\sigma}_\zeta = \text{bvec}\{S_{\lambda,k}\} \quad (\textbf{EMSE}) \tag{3.88b}$$

so that

$$\eta_k = b^T (I - \overline{F})^{-1} \text{bvec}\{S_{q,k}\} \quad (\textbf{MSD}) \tag{3.89a}$$

$$\zeta_k = b^T (I - \overline{F})^{-1} \text{bvec}\{S_{\lambda,k}\} \quad (\textbf{EMSE}) \tag{3.89b}$$

3.5 Results and Discussion

The simulation parameters defined in section 2.4 are taken here. Fig.3.1 shows the network topology. Here we have chosen the number of nodes $N = 7$. The network statistics are given in Fig.3.2(a) and 3.2(b). Here the back ground noise is white and Gaussian with variance $\sigma_v^2 = 10^{-3}$, and we generated the data using (3.16). In order to generate the performance curves, 50 independent experiments were performed and averaged.

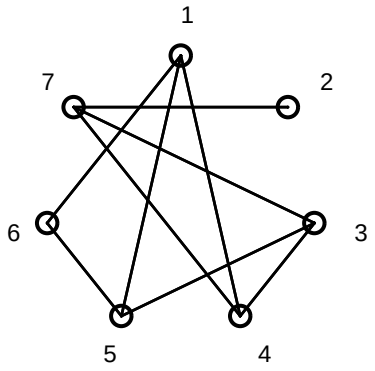
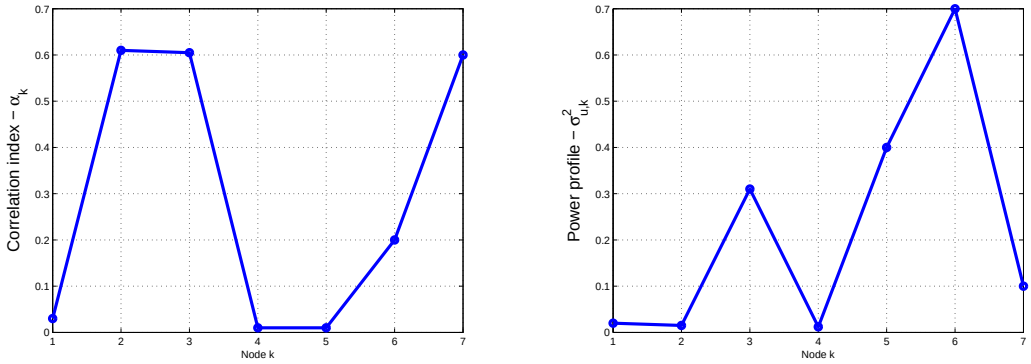


Figure 3.1: Network Topology



(a) (b)

Figure 3.2: Network statistics. (a) Network co-relation index pernode. (b) Regressor power profile.

The curves are generated by running the network learning process for 10,000 iterations. The quantities of interest are then obtained by averaging the last 5000 samples of the corresponding learning curves. The global mean-square deviation(MSD) curve is given in Fig.3.3. This is obtained by averaging $E \parallel \tilde{\Psi}_k^{(j-1)} \parallel^2$ across all the nodes and over 50 experiments. Similarly the global excess mean-square error(EMSE) curve is given in Fig.3.4. This is obtained by averaging $E \parallel \mathbf{e}_{a,k}(j) \parallel^2$, where $\mathbf{e}_{a,k}(j) = \mathbf{x}_{k,j}\tilde{\Psi}_k^{(j-1)}$ across all the nodes and over 50 experiments. The global quantities are the contributions of individual nodes and it is obtained simply by taking the mean of all individual nodes. The local MSD evaluated at node 1 is given in Fig. 3.5 and at node 5 is given in Fig. 3.5. Similarly the local EMSE evaluated at nodes 1 and 5 are depicted in Fig. 3.7.

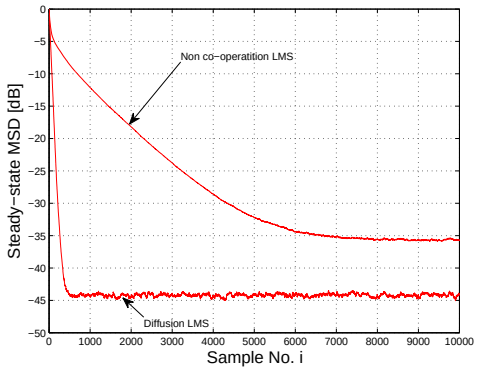


Figure 3.3: Global mean-square deviation(MSD) curve for diffusion and non-cop. way of operation.

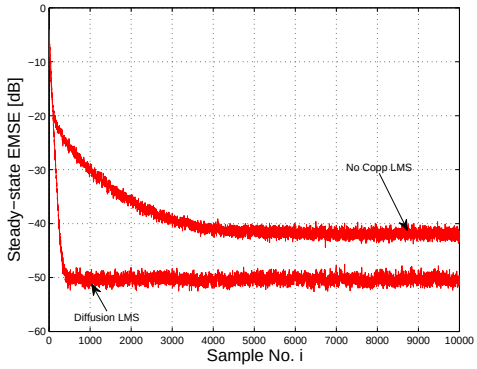


Figure 3.4: Global excess mean-square deviation(EMSE) curve for diffusion and non-cop. way of operation.

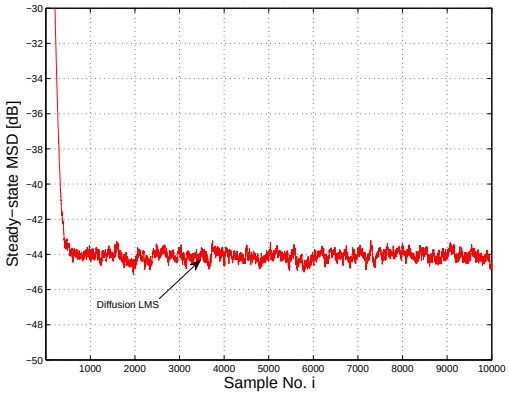


Figure 3.5: Local mean-square deviation(MSD) curve at node 1

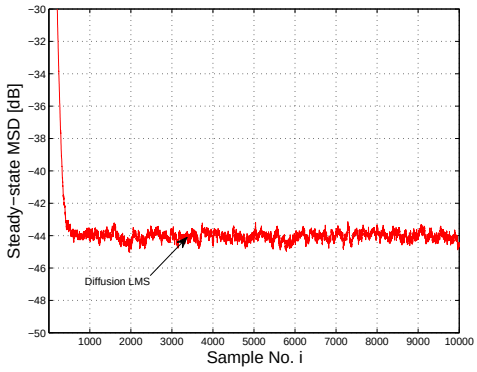


Figure 3.6: Local mean-square deviation(MSD) curve at node 5

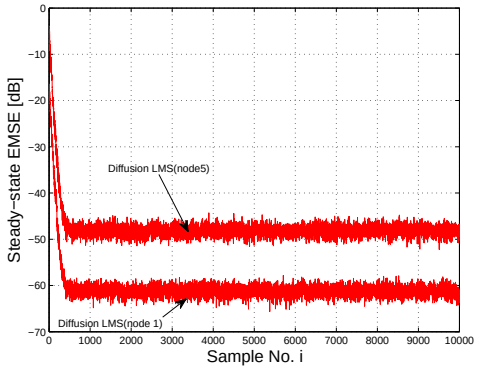


Figure 3.7: Local EMSE at nodes 1 and 5 for the same network

3.6 Conclusion

The mathematical analysis and the simulation results show that cooperation improves performance by reducing computation and communication resources. It has a stabilizing effect on the network. Filters can be designed using local information to achieve local stability and diffusion protocols can be implemented to improve the global performance. Closed-form expressions for global and local mean and mean-square performance have been derived, matching very well the simulations that have been carried out.

Chapter 4

Distributed Block Least Mean Square Algorithm

4.1 Introduction

Distributed signal processing can be integrated with the concept of block LMS to obtain distributed block LMS algorithm. Here diffusion protocol [3] is considered in which nodes from the same neighborhood are allowed to communicate with each other after a block of data. Hence the number of communications among the neighbor nodes decrease which is advantageous in any wireless sensor network.

In this approach block LMS is used instead of the simple LMS in each local adaptive filter. Each node updates its estimate for each block of data, then interact with neighboring node for diffusion. In this case the number of communications between the nodes reduce to block length times than the general LMS. The contribution of this chapter is to propose a block diffusion adaptive network, and to study its performance for Gaussian data. This analysis can also be extended to distributed incremental block LMS algorithm.

4.2 Block Diffusion LMS

The block Least-mean square (BLMS) algorithm is described in [12] and its performance is studied in [13],[14],[15]. In adaptive filtering the filter coefficients are adjusted once per each block in accordance with a generalized least mean square(LMS) algorithm. Analysis shows [12] that the block adaptive filter permits faster implementation while maintaining equivalent performance to that of widely used LMS adaptive filter.

We would like to estimate an $M \times 1$ unknown vector $\mathbf{w}^{\circ}$ from measurements collected at N nodes spread over a network (see Fig. 1). Each node k has access to time realizations $\{d_k(i), \mathbf{u}_{k,i}\}$ of zero mean random data $\{d_k, \mathbf{u}_k\}, k = 1, 2, 3, ...N$, with $d_k(i)$ is scalar measurement and $\mathbf{u}_{k,j}$ a $1 \times M$ regression row vector, both at time j is given as.

$$\mathbf{u}_{k,i} = [u_k(i), u_k(i-1), \dots u_k(i-M+1)]$$

In this paper we realize adaptive filters at each node by block wise processing of the data in order to take computation and communication among neighbors is major advantage. Here the adaptive algorithm must allow to calculate whole block of outputs without modifying the weights. Thus a block adaptive filter adjusts weight once per block of data. Let L represents the block length, the following variables are defined for the block processing of data at each node as

$$\mathbf{X}_{k,j} = [\mathbf{u}_{k,(j-1)L+1}, \mathbf{u}_{k,(j-1)L+2}, \dots, \mathbf{u}_{k,jL}]^T$$

is the block input data for node k and

$$\mathbf{y}_{k,j} = [d_k((j-1)L+1), d_k((j-1)L+1), \ldots, d_k(jL)]^T$$

is the output corresponding to that block of input data. We collect the regression and measurement data across all nodes into two global matrices. We drop the time index for compactness of notation.

$$\mathbf{X}_c = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N]^T, \qquad \mathbf{U}_c = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]^T \tag{4.1a}$$

$$\mathbf{y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N]^T, \qquad \mathbf{d} = [d_1, d_2, \ldots, d_N]^T \tag{4.1b}$$

Let $\varepsilon_{k,j}$ is block of errors of node k and is defined as

$$\epsilon_{k,j} = \mathbf{y_{k,j}} - \mathbf{X_{k,j}}\mathbf{w}$$

therefore the global error vector

$$\epsilon = [\epsilon_1, \epsilon_2, \ldots, \epsilon_N]^T = \mathbf{y} - \mathbf{X}_c\mathbf{w}$$

Now we can define the block mean square error(BMSE) as

$$BMSE = \frac{1}{L}E[\epsilon^T\epsilon] \tag{4.2}$$

$$= \frac{1}{L}E[(\mathbf{y} - \mathbf{X_b}\mathbf{w})^\mathbf{T}(\mathbf{y} - \mathbf{X_b}\mathbf{w})]$$

$$= \frac{1}{L}[E[\mathbf{y}^T\mathbf{y}] - E[\mathbf{y}^T\mathbf{X}_b]\mathbf{w} - \mathbf{w}^T E[\mathbf{X}_b^T\mathbf{y}] - \mathbf{w}^T E[\mathbf{X}_b^T\mathbf{X}_b]\mathbf{w}]$$

Proposition 1 *If the input regression data u is Gaussian and defined with correlation function* $r(l) = \sigma^2\alpha^{|l|}$ *in the covariance matrix, where* α *is the correlation index and* σ^2 *is the variance of the input regression data, then let us define some variables as* :

- $\mathbf{u}$ *is* $1 \times M$ *regressor data*
- $\mathbf{X} = \left[\mathbf{u}_{(j-1)L+1}, \mathbf{u}_{(j-1)L+2}, \ldots, \mathbf{u}_{jL}\right]^T$
- $\mathbf{X}_c = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N]^T$

- $\mathbf{U}_c = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]^T$
- $R_x = E[\mathbf{X}^T\mathbf{X}]$ *and* $R_X = E[\mathbf{X}_c^T\mathbf{X}_c]$
- *similarly* $R_u = E[\mathbf{u}^T\mathbf{u}]$ *and* $R_U = E[\mathbf{U}_c^T\mathbf{U}_c]$

then It can be shown that

$$R_X = LR_U$$

and

$$R_x = LR_u$$

Proof : Let us define the regression data as

$$u(i) = \alpha \cdot u(i-1) + \beta \cdot z(i), \qquad i > -\infty$$

Here, $\alpha \in [0, 1)$ *is the correlation index and* $z(i)$ *is a l spatially independent white Gaussian process with unit variance and* $\beta = \sqrt{\sigma_u^2 \cdot (1 - \alpha^2)}$*. The regressor power profile* $\{\sigma_u^2\} \in (0, 1]$*. The resulting regressors have Toeplitz covariance with co-relation sequence* $r(i) = \sigma_u^2 \cdot (\alpha)^{|i|}, i = 0, 1, 2, \ldots, M-1$*.*

$$u = [u_3, u_2, u_1]$$

$$x = \begin{bmatrix} u_3 & u_2 & u_1 \\ u_4 & u_3 & u_2 \\ u_5 & u_4 & u_3 \end{bmatrix}$$

Where u is the input regressor data vector and x is the block of input regressor data. For our simplicity here we have chosen the length of block and the size of filter both are 3. So, now we can define the auto-correlation function $R_u = E[u^T u]$ *and* $R_x = E[x^T x]$*. These are computed as*

$$R_u = \begin{bmatrix} u_3^2 & u_3u_2 & u_3u_1 \\ u_2u_3 & u_2^2 & u_2u_1 \\ u_1u_3 & u_1u_2 & u_1^2 \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_u^2\alpha & \sigma_u^2\alpha^2 \\ \sigma^2\alpha^2 & \sigma^2 & \sigma_u^2\alpha \\ \sigma_u^2\alpha^2 & \sigma_u^2\alpha & \sigma_u^2 \end{bmatrix}$$

$$R_x = E[x^T x] = \begin{bmatrix} 3\sigma_u^2 & 3\sigma_u^2\alpha & 3\sigma_u^2\alpha^2 \\ 3\sigma^2\alpha^2 & 3\sigma^2 & 3\sigma_u^2\alpha \\ 3\sigma_u^2\alpha^2 & 3\sigma_u^2\alpha & 3\sigma_u^2 \end{bmatrix} = 3R_u$$

This result can be generalized to any input regressor data length defined with some correlation function. If the input data is independent Gaussian variable with no correlation, the diagonal elements of the correlation matrix are zero. So, it can be easily verified that $R_x = R_u$.

Now we can define the autocorrelation matrix $\mathrm{R}_x = E[\mathbf{X}_c^T \mathbf{X}_c]$ and it can be shown that $\mathrm{R}_x = L\mathrm{R}_u$ where $\mathrm{R}_u = E[\mathbf{U}_c^T \mathbf{U}_c]$ is the autocorrelation matrix without incorporating block concept in the input data. Similarly we can also define the cross-correlation matrix $\mathrm{R}_{yx} = E[\mathbf{X}_c^T \mathbf{y}]$ and we can also expressed as $\mathrm{R}_{xy} = L\mathrm{R}_{du}$ where $\mathrm{R}_{du} = E[\mathbf{U}_c^T \mathbf{d}]$ is the cross correlation vector without doing in block. This is given as $Proposition - 1$. Therefore the BMSE is given by

$$BMSE = \frac{1}{L}[E[\mathbf{d}^T\mathbf{d}] - \mathrm{R}_{du}^T\mathbf{w} - \mathbf{w}^T\mathrm{R}_{du} - \mathbf{w}^T\mathrm{R}_u\mathbf{w}]$$

so we seek to minimize the equation

$$\min_{\mathbf{w}} E \parallel \mathbf{d} - \mathbf{U}_c\mathbf{w} \parallel^2 \tag{4.3}$$

so solution for this is[12]

$$\mathbf{w}^\circ = \mathrm{R}_u^{-1}\mathrm{R}_{du} \tag{4.4}$$

For later reference, we also introduce the block diagonal matrix

$$\mathbf{X} = \mathrm{diag}\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\} \qquad (LN \times NM) \tag{4.5}$$

and

$$\mathbf{Q} = [\mathrm{I}_M, \mathrm{I}_M, \ldots, \mathrm{I}_M]^T \qquad (NM \times M) \tag{4.6}$$

where I_M is the $M \times M$ identity matrix which is related X_c as

$$\mathbf{X}_c = \mathbf{X}\mathbf{Q} \tag{4.7}$$

4.2.1 Diffusion Block LMS algorithm

The analysis of the diffusion block LMS is carried out in the same line with diffusion LMS discussed in section 3.2.1. Analysis and simulation will show that the diffusion block LMS algorithm leads to a robust distributed adaptive system that achieve same error level with less number of communications compared to Diffusion LMS in [3].

The proposed block diffusion strategy can therefore be described in general form similar to the diffusion strategy defined in (3.14):

$$\phi_k^{(j-1)} = f_k\left(\psi_l^{(j-1)}; l \in \mathcal{N}_{k,j-1}\right) \tag{4.8}$$

$$\psi_k^{(j)} = \phi_k^{(j-1)} + \frac{\mu_k}{L} \sum_{q=(j-1)L+1}^{jL} \mathbf{u}_{k,q}^T \left(d_k(q) - \mathbf{u}_{k,q}\phi_k^{(j-1)}\right) \tag{4.9}$$

for local step size μ_k.

4.3 Network Global Model

We introduce the global quantities as

$$\psi^j = [\psi_1^{(j)}, \ldots, \psi_N^{(j)}]^T, \qquad \phi^j = [\phi_1^{(j)}, \ldots, \phi_N^{(j)}]^T$$

$$\mathbf{X}_j = \text{diag}\{\mathbf{X}_{1,j}, \ldots, \mathbf{X}_{N,j}\}, \qquad \mathbf{y}_j = [\mathbf{y}_{1,j}, \ldots, \mathbf{y}_{N,j}]^T$$

Let us define another variable as

$$D = \text{diag}\{\mu_1 \mathrm{I}_M, \mu_2 \mathrm{I}_M, \ldots, \mu_N \mathrm{I}_M\} \qquad (NM \times NM) \tag{4.10}$$

be a diagonal matrix collecting the local step sizes. The measurements are assumed to obey a traditional model of the form [6],[5], [10]

$$\mathbf{y}_{k,j} = \mathbf{X}_{k,j}\mathbf{w}^\circ + \mathbf{v}_{k,j} \tag{4.11}$$

where $\mathbf{v}_{k,j}$ is background noise, assumed independent over time and space and with variance $\sigma_{v,k}^2$. Linear model of the (4.11) are able to capture or approximate many input output relations for estimation, we can write the global linear model as

$$\mathbf{y}_j = \mathbf{X}_j w^{(\circ)} + \mathbf{v}_j \tag{4.12}$$

where $\mathbf{w}^{(\circ)} = Q\mathbf{w}^\circ$ and

$$\mathbf{v}_j = [\mathbf{v}_{1,j}, \ldots, \mathbf{v}_{N,j}]^T \qquad (LN \times 1)$$

With these relations, the algorithm can be written in global form:

$$\begin{aligned} \phi^{j-1} &= G\psi^{j-1} \\ \psi^{j} &= \phi^{j-1} + \frac{1}{L} D\mathbf{X}_j^T(\mathbf{y}_j - \mathbf{X}_j\phi^{j-1}) \end{aligned} \tag{4.13}$$

or in a more compact for it can be written as

$$\psi^{j} = G\psi^{j-1} + \frac{1}{L} D\mathbf{X}_j^T(\mathbf{y}_j - \mathbf{X}_j G\psi^{j-1}) \tag{4.14}$$

where $G = C \otimes I_M$ is the $NM \times NM$ transition matrix and C is the $N \times N$ diffusion combination matrix with entries $[c_{kl}]$.

Combiner Coefficient

For further analysis the combiner coefficient used in section 3.3 is introduced here.

4.4 Performance Analysis

The assumptions and energy based approach discussed in section 3.4 are used here for performance analysis of diffusion block LMS algorithm.

4.4.1 Mean Transient Analysis

First we examine the mean behavior of the network and will show it is similar to the diffusion LMS network. Thus introduce the global weighted error vector

$$\tilde{\psi}^{j} = \mathbf{w}^{\circ} - \psi^{j} \tag{4.15}$$

Since, $G\mathbf{w}^{(\circ)} = \mathbf{w}^{(\circ)}$, using global data model (4.12) and subtracting $\mathbf{w}^{\circ}$ from the left hand side and $G\mathbf{w}^{(\circ)}$ from the right side of (4.14), we get

$$\tilde{\psi}^{j} = G\tilde{\psi}^{j-1} - \frac{1}{L} DX_j^T \left(X_j G\mathbf{w}^{(\circ)} + v_j - X_j G\tilde{\psi}^{j-1} \right)$$

We can write this in compact form as

$$\tilde{\psi}^{j} = \left(\mathrm{I}_{NM} - \frac{1}{L} DX_j^T X_j \right) G\tilde{\psi}^{j-1} - DX_j^T \mathbf{v}_j \tag{4.16}$$

Assuming the big assumption and spatial independence of the regression data $\{\mathbf{X}_{k,j}\}$ and taking the expectation on both sides of the above equation (4.16) gives to

$$E[\tilde{\psi}^j] = \left(\mathrm{I}_{NM} - \frac{1}{L} D\mathrm{R}_x\right) GE[\tilde{\psi}^{j-1}] \tag{4.17}$$

where we can view as $\mathrm{R}_x = \mathrm{diag}\{\mathrm{R}_{x,1}, \mathrm{R}_{x,2}, \ldots, \mathrm{R}_{x,N}\}$ is block diagonal and $\mathrm{R}_{x,k} = E[\mathbf{X}_{k,j}^T \mathbf{X}_{k,j}] = LE[\mathbf{u}_{k,j}^T \mathbf{u}_{k,j}] = L\mathrm{R}_u$. Hence the equation reduces to

$$E[\tilde{\psi}^j] = (\mathrm{I}_{NM} - D\mathrm{R}_u)\, GE[\tilde{\psi}^{j-1}] \tag{4.18}$$

The mean analysis of the above equation (4.18) is given in [3] and it can be also shown that block diffusion protocol defined above has a stabilizing effect on the network.

4.4.2 Mean-Square Transient Analysis

The expression for mean square deviation (MSD) and excess mean square deviation (EMSE) is derived here.

We can define the block of local output estimation error at node k for block j as

$$\mathbf{e}_{k,j} = \mathbf{y}_{k,j} - \mathbf{X}_{k,j}\phi_k^{(j-1)} \tag{4.19}$$

Now the global error vector can be written by collecting the error across the network as:

$$\mathbf{e}_j = [\mathbf{e}_{1,j}, \mathbf{e}_{2,j}, \ldots, \mathbf{e}_{N,j}]^T \qquad (NL \times 1)$$

so that we can write

$$\mathbf{e}_j = \mathbf{y}_j - \mathbf{X}_j G\Psi^{(j-1)} = \mathbf{X}_j G\tilde{\Psi}^{(j-1)} + \mathbf{v}_j = \mathbf{e}_{a,j}^G + \mathbf{v}_j \tag{4.20}$$

where

$$\mathbf{e}_{a,j}^G = \mathbf{X}_j G\tilde{\Psi}^{(j-1)} \tag{4.21}$$

Introducing global *a priori* and *a posteriori* weighted estimation errors:

$$\mathbf{e}_{a,j}^{D\Sigma G} = \mathbf{X}_j D\Sigma G\tilde{\psi}^{(j-1)} \qquad \text{and} \qquad \mathbf{e}_{p,j}^{D\Sigma} = \mathbf{X}_j D\Sigma\tilde{\psi}^{(j-1)} \tag{4.22}$$

for some arbitrary $NM \times NM$ matrix $\Sigma \geq 0$. The freedom in selecting Σ enable us to characterize MSD and EMSE performance of the network. Now we redefine the global

weight error vector as

$$\begin{aligned}\tilde{\psi}^j &= \mathbf{w}^\circ - \Psi^j \\ &= G\mathbf{w}^{(\circ)} - G\Psi^{j-1} - \frac{1}{L}D\mathbf{X}_j^T(\mathbf{y}_j - \mathbf{X}_j G\Psi^{j-1}) \\ &= G\tilde{\Psi}^{j-1} - \frac{1}{L}D\mathbf{X}_j^T\mathbf{e}_j \end{aligned} \tag{4.23}$$

Multiplying $\mathbf{X}_j D\Sigma$ on both sides of (4.23)

$$\mathbf{e}_{p,j}^{D\Sigma} = \mathbf{e}_{a,j}^{D\Sigma G} - \frac{1}{L}\| \mathbf{X}_j \|_{D\Sigma D}^2 \mathbf{e}_j \tag{4.24}$$

therefore the global error can be written as:

$$\mathbf{e}_j = \frac{\mathbf{e}_{a,j}^{D\Sigma G} - \mathbf{e}_{p,j}^{D\Sigma}}{\frac{1}{L}\| \mathbf{X}_j \|_{D\Sigma D}^2} \tag{4.25}$$

using the above equation (4.25) in (4.23), we get

$$\begin{aligned}\tilde{\Psi}^j &= G\tilde{\Psi}^{j-1} - \frac{1}{L}D\mathbf{X}_j^T \frac{\mathbf{e}_{a,j}^{D\Sigma G} - \mathbf{e}_{p,j}^{D\Sigma}}{\frac{1}{L}\| \mathbf{X}_j \|_{D\Sigma D}^2} \\ \tilde{\Psi}^j + \frac{D\mathbf{X}_j \mathbf{e}_{p,j}^{D\Sigma}}{\| \mathbf{X}_j \|_{D\Sigma D}^2} &= G\tilde{\Psi}^{j-1} + \frac{D\mathbf{X}_j \mathbf{e}_{a,j}^{D\Sigma G}}{\| \mathbf{X}_j \|_{D\Sigma D}^2} \end{aligned} \tag{4.26}$$

Equating the weighted norm of both sides of (4.26), we find that the cross term cancel out, and we end up only the energy terms,i.e.,

$$\| \tilde{\Psi}^j \|_\Sigma^2 + \frac{| \mathbf{e}_{a,j}^{D\Sigma G} |^2}{\| \mathbf{X}_j \|_{D\Sigma D}^2} = \| \tilde{\Psi}^{j-1} \|_{G^T\Sigma G}^2 + \frac{| \mathbf{e}_{p,j}^{D\Sigma} |^2}{\| \mathbf{X}_j \|_{D\Sigma D}^2} \tag{4.27}$$

Equation (4.27) is a $space - time$ version of the weighted energy conservation relation. This is an exact relation shows that how energies of several error variables are related to each other in space and time. Up to now (4.27), no approximation are used.

Variance Relation

Replacing $\mathbf{e}_{p,j}^{D\Sigma}$ by using its definition in (4.24), we get

$$\| \tilde{\psi}^j \|_\Sigma^2 = \| \tilde{\psi}^{j-1} \|_{G^T\Sigma G}^2 - \frac{1}{L}\mathbf{e}_j^T \mathbf{e}_{a,j}^{D\Sigma G} - \frac{1}{L}(\mathbf{e}_{a,j}^{D\Sigma G})^T \mathbf{e}_j + \frac{1}{L^2}\| \mathbf{X}_j \|_{D\Sigma D}^2 \mathbf{e}_j^T \mathbf{e}_j \tag{4.28}$$

Using error relation (4.20) and taking expectation on both sides, we get

$$E\| \tilde{\psi}^j \|_{\Sigma}^2 = E\| \tilde{\psi}^{j-1} \|_{G^T\Sigma G}^2 - \frac{1}{L}E\| \tilde{\psi}^{j-1} \|_{G^T\Sigma D\mathbf{X}_j^T\mathbf{X}_j G}^2 - \frac{1}{L}E\| \tilde{\psi}^{j-1} \|_{G^T\mathbf{X}_j^T\mathbf{X}_j D\Sigma G}^2 + \frac{1}{L^2}E\| \tilde{\psi}^{j-1} \|_{G^T\mathbf{X}_j^T\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\mathbf{X}_j G}^2 + \frac{1}{L^2}E\mathbf{v}_j^T\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\mathbf{v}_j \quad (4.29)$$

Now, using the relation $\| x \|_A^2 + \| x \|_B^2 = \| x \|_{A+B}^2$, the previous equation can be written more compactly as

$$E\| \tilde{\psi}^j \|_{\Sigma}^2 = E\| \tilde{\psi}^{j-1} \|_{\mathbf{\Sigma}'}^2 + \frac{1}{L^2}E\mathbf{v}_j^T\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\mathbf{v}_j \quad (4.30)$$

where

$$\mathbf{\Sigma}' = G^T\Sigma G - \frac{1}{L}G^T\Sigma D\mathbf{X}_j^T\mathbf{X}_j G - \frac{1}{L}G^T\mathbf{X}_j^T\mathbf{X}_j D\Sigma G - \frac{1}{L^2}G^T\mathbf{X}_j^T\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\mathbf{X}_j G \quad (4.31)$$

Continuing the analysis in the same way as done in section 3.4.2, equations (4.30)-(4.31) reduces to

$$E\| \tilde{\Psi}^j \|_{\Sigma}^2 = E\| \tilde{\psi}^{j-1} \|_{\Sigma'}^2 + \frac{1}{L^2}E\mathbf{v}_j^T\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\mathbf{v}_j \quad (4.32)$$

$$\Sigma' = G^T\Sigma G - \frac{1}{L}G^T\Sigma DE(\mathbf{X}_j^T\mathbf{X}_j)G - \frac{1}{L}G^T E(\mathbf{X}_j^T\mathbf{X}_j)D\Sigma G + \frac{1}{L^2}G^T E(\mathbf{X}_j^T\mathbf{X}_j)D\Sigma DE(\mathbf{X}_j^T\mathbf{X}_j)G \quad (4.33)$$

Using *proposition* -1 in equation (4.33) i.e.$E\mathbf{X}_j^T\mathbf{X}_j = LE\mathbf{U}_j^T\mathbf{U}_j$,

$$\Sigma' = G^T\Sigma G - G^T\Sigma DE(\mathbf{U}_j^T\mathbf{U}_j)G - G^T E(\mathbf{U}_j^T\mathbf{U}_j)D\Sigma G + G^T E(\mathbf{U}_j^T\mathbf{U}_j)D\Sigma DE(\mathbf{U}_j^T\mathbf{U}_j)G \quad (4.34)$$

Now we can proceed for 2nd term of equation (4.32) by exploiting some properties of expectation and trace as follows [16], [17]:

$$\begin{aligned}\frac{1}{L^2}E\mathbf{v}_j^T\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\mathbf{v}_j &= \frac{1}{L^2}E\left[\parallel \mathbf{X}_j \parallel^2_{D\Sigma D}\mathbf{v}_j^T\mathbf{v}_j\right] \\ &= \frac{1}{L^2}E\left[\mathrm{tr}\left[\mathbf{X}_j D\Sigma D\mathbf{X}_j^T\right]\right]E\left[\mathbf{v}_j^T\mathbf{v}_j\right] \\ &= \frac{1}{L^2}\mathrm{tr}\left[D\Sigma DE\left[\mathbf{X}_j\mathbf{X}_j^T\right]\right]LE\left[\mathbf{V}_j^T\mathbf{V}_j\right] \\ &= \frac{L}{L^2}\mathrm{tr}\left[D\Sigma DE\left[\mathbf{U}_j\mathbf{U}_j^T\right]\right]LE\left[\mathbf{V}_j^T\mathbf{V}_j\right] \\ &= E\left[\mathbf{V}_j^T\mathbf{U}_j D\Sigma D\mathbf{U}_j^T\mathbf{V}_j\right] \end{aligned} \tag{4.35}$$

where $\mathbf{V}_j$ is the noise vector in absence of block concept. Assume that the noise is stationary and Gaussian. Therefore we can write equations (4.32) and (4.33) as:

$$E\parallel \tilde{\psi}^j \parallel^2_{\Sigma} = E\parallel \tilde{\psi}^{j-1} \parallel^2_{\Sigma'} + \frac{1}{L^2}E\mathbf{V}_j^T\mathbf{U}_j D\Sigma D\mathbf{U}_j^T\mathbf{V}_j \tag{4.36}$$

$$\begin{aligned}\Sigma' = G^T\Sigma G - G^T\Sigma DE(\mathbf{U}_j^T\mathbf{U}_j)G - G^T E(\mathbf{U}_j^T\mathbf{U}_j)D\Sigma G \\ + G^T E(\mathbf{U}_j^T\mathbf{U}_j)D\Sigma DE(\mathbf{U}_j^T\mathbf{U}_j)G\end{aligned} \tag{4.37}$$

The above equations are exactly same as equations (3.42) and (3.43) in section 3.4.2. Further mathematical analysis can be carried out as discussed in section 3.4.2. But in adaptive diffusion block LMS, the local step-size should be L(the block size) times the local step-size chosen in diffusion LMS. This results in a faster(L times) convergence [12].

4.5 Results and Discussion

Here all simulations were carried out using regressors with shift invariance structure to cope with realistic scenario. Therefore the regressor are filled up as

$$\mathbf{u}_{k,i} = [u_k(i), u_k(i-1), \ldots, u_k(i-M+1)]^T \tag{4.38}$$

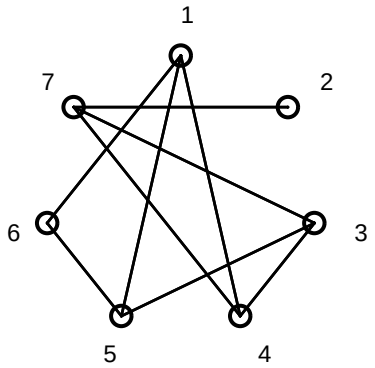


Figure 4.1: Network Topology

and in the block regressor is given as (where $L = 3$ and $M = 3$)

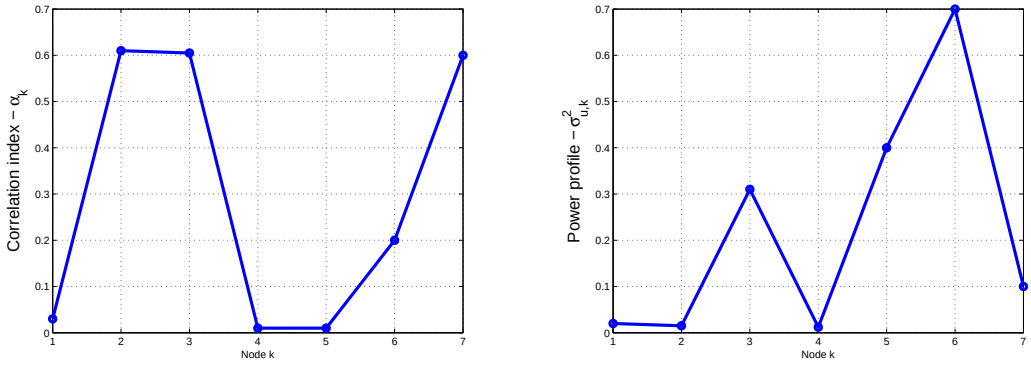
$$\mathbf{X}_{k,1} = \begin{pmatrix} u_k(1) & 0 & 0 \\ u_k(2) & u_k(1) & 0 \\ u_k(3) & u_k(2) & u_k(1) \end{pmatrix}$$
$$\mathbf{X}_{k,2} = \begin{pmatrix} u_k(4) & u_k(3) & u_k(2) \\ u_k(5) & u_k(4) & u_k(3) \\ u_k(6) & u_k(5) & u_k(4) \end{pmatrix} \tag{4.39}$$

The simulation parameters defined in section 3.5 are taken here for comaprison purpose.

Fig.4.1 shows the network topology. Here the number of nodes are chosen as $N = 7$. The network statistics are given in Fig.4.2(a) and 4.2(b).

4.5.1 Choice of Block Length

This algorithm is valid for any block length greater than one [12]. Preferably the L is kept equal to M. For L greater than M, the gradient estimate is computed over L input points wheras the filter uses less inputs resulting in redundant operation. For L less than M, the filter length is larger than the input block, which is a wastage of filter inputs. Therefore the optimum choice for block length is eual to the filter size. Similar to section 3.5 the curves are generated by running the network learning process for 10,000 samples of data i.e. $\left(\frac{10000}{\text{Length of filter}}\right)$ blocks. The global mean-square deviation(MSD) curve is given in Fig.4.3. Similarly the global excess mean-square error(EMSE) curve is given in



(a) (b)

Figure 4.2: Network statistics. (a) Network co-relation index pernode. (b) Regressor power profile.

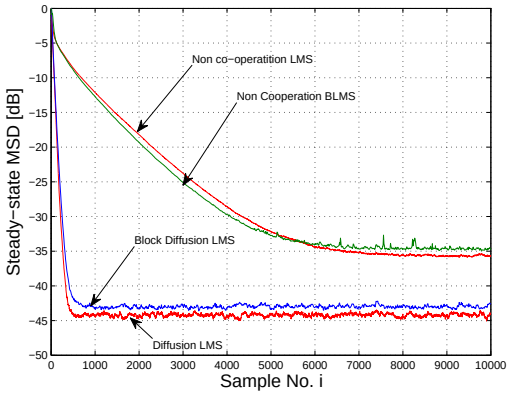


Figure 4.3: Global mean-square deviation(MSD) curve for diffusion and non-cop. way of operation.

Fig.4.4. If we update the weights after L numbers of data points and then communicate for local diffusion, the number of communications between neighbors reduces to $frac1L$ times the number of communications in diffusion LMS where the weights are updated after each sample of data.

4.5.2 Local Node performance

The global quantities are the contributions of individual nodes and it is obtained simply by taking the mean of all individual nodes. The mathematical expression for local MSD and EMSE is also given in [3]. In diffusion block LMS the expression remains the same as equation (3.89) because these are derived from the same variance relation. Simulation

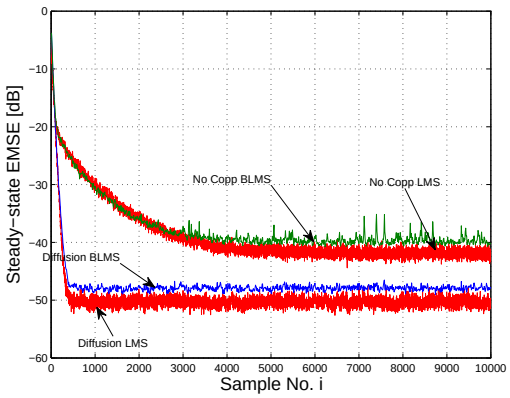
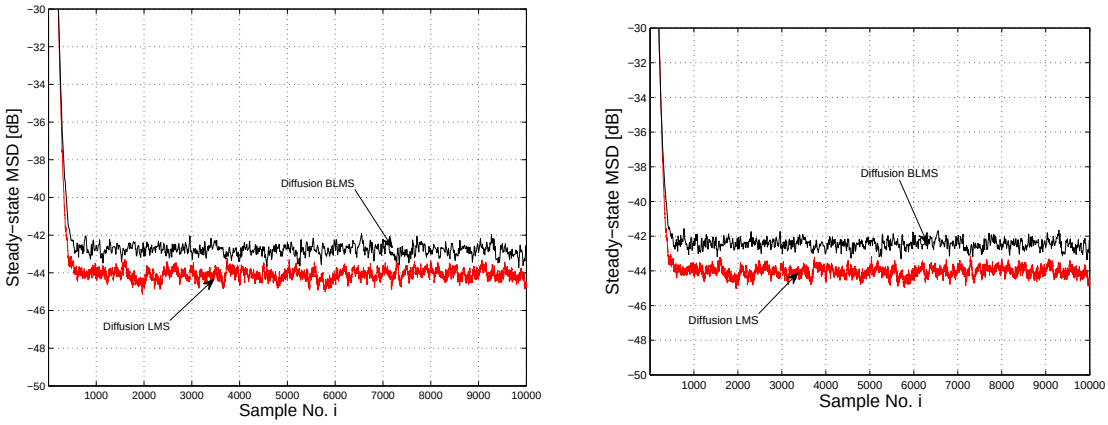


Figure 4.4: Global excess mean-square deviation(EMSE) curve for diffusion and non-cop. way of operation.

results of block diffusion LMS algorithm are compared with the results of diffusion LMS. The local MSD evaluated at node 1 is given in Fig. 4.5(a) and at node 5 is given in Fig. 4.5(b). Similarly the local EMSE evaluated at nodes 1 and 5 are depicted in Fig. 4.6. The convergence speed of MSD and EMSE in both approaches are nearly same. In diffusion block LMS the number of communications between neighbors decrease at the cost of degradation in performance.



(a) (b)

Figure 4.5: Local mean-square deviation(MSD) comparison between block Diffusion LMS and Diffusion LMS. (a) MSD curve at node 1. (b) MSD curve at node 5.

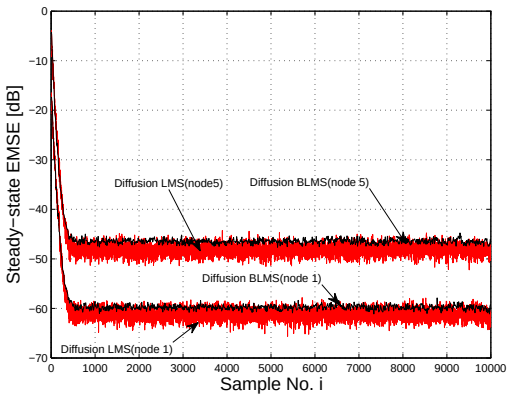


Figure 4.6: Local EMSE at nodes 1 and 5 for the same network

4.6 Conclusion

This chapter dealt with a new distributed LMS algorithm in which the block concept has been incorporated into the diffusion LMS algorithm discussed in chapter 3. The mathematical analysis and the simulation results show that the performance of the diffusion block LMS algorithm is nearly same as diffusion LMS. The number of communications between neighbor nodes decrease to $frac1L$ times the number of communications in diffusion LMS. This approach is preferred in those applications where there is a severe bound on communication resources.

Chapter 5

Robust Estimation in Wireless Sensor Network

5.1 Introduction

The main challenges in estimating parameters in a sensor network are link failure and impulsive noise. Under such adverse conditions the conventional square error cost function based learning algorithms offer poor estimation performance. Several methods for estimation problems are proposed in the literature which exploit the physical behavior of the event being measured. The most important characteristic of any event is the correlation between the measurements at different nodes. Recently, several distributed optimization algorithms based on gradient search have been proposed. Due to the large amount of complexity in assuring convergence of distributed gradient search algorithms, the objective function is assumed to be additive and convex.

Distributed signal processing deals with the extraction of information from local data collected at nodes that are distributed over a geographical area. Each node in a network records noisy observations related to the parameter to be estimated. The nodes would then interact with their neighbors in a certain manner, according to the network topology, either in an incremental [4] or by diffusion [3] approach. A network is more efficient if it requires less communication between nodes to estimate the parameter vector [11], [1].

5.1.1 Distributed Optimization Techniques

As discussed in chapter 2 and chapter 3, distributed wireless sensor networks are characterized by two modes of cooperation i.e. incremental or diffusion. Here we focus on the incremental mode of cooperation. Consider a network consisting of N nodes. Each node has access to a local temperature measurement T_i. The objective is to provide each node with information about the average temperature $\hat{T}$. Averages can be viewed as the values minimizing quadratic cost functions. Quadratic optimization problems have solutions which are linear functions of the data. A simple accumulation of parameter estimate leads to a solution. General optimization problems can often be solved using this simple, distributed algorithms.

In general, an optimization problem can be expressed as:

$$f(\theta) = \frac{1}{N} \sum_{i=1}^{N} f_i(\theta) \tag{5.1}$$

where θ is the parameter to be estimated, and $f(\theta)$ is the cost function which can be expressed as a sum of N *local* cost functions $\{f_i(\theta)\}_N^{i=1}$ in which $f_i(\theta)$ only depends on

the data measured at sensor i and is given by,

$$f_i(\theta) = \frac{1}{M}\sum_{j=1}^{M}(x_{i,j} - \theta)^2 \tag{5.2}$$

where $x_{i,j}$ is the j-th measurement of i-th sensor.

Hence putting the value of $f_i(\theta)$ from (5.2) into (5.1),

$$f(\theta) = \frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}(x_{i,j} - \theta)^2 \tag{5.3}$$

In the proposed approach, an estimate of the parameter θ is passed from node to node. Each node updates the parameter to reduce its local cost (5.2) and then passes the updated parameter to the next node. The flow of information from first node to the last node forms a single cycle. Several cycles through the network are required to obtain a solution. These distributed algorithms can be viewed as incremental subgradient optimization procedures, and the number of cycles required to obtain a good solution can be characterized theoretically. If M and N are large, then a high quality estimate can be obtained using a distributed optimization algorithm with less energy and communications than the centralized approach.

5.2 Decentralized Incremental Optimization

For a convex differentiable function, $f : \Theta \to \mathbb{R}$, the following inequality for the gradient of f at a point θ_0 holds for all $\theta \in \Theta$:

$$f(\theta) \geq f(\theta_0) - (\theta - \theta_0)^T \nabla f(\theta_0)$$

In general, for a convex function *f*, a subgradient of f at θ_0 (observing that f may not be differentiable at θ_0) is any direction g such that

$$f(\theta) \geq f(\theta_0) - (\theta - \theta_0)^T g \tag{5.4}$$

and the subdifferential of f at θ_0, denote $\partial f(\theta_0)$, is the set of all subgradients of f at θ_0). Note that if f is differentiable at θ_0, then $\partial f(\theta_0) \equiv \{\nabla f(\theta_0)\}$; *i.e.*, the gradient of f at θ_0 is the only direction satisfying (5.4).

Here a network of N sensors is considered in which each sensor collects M measure-

ments. Let $x_{i,j}$ denote the j-th measurement taken at the i-th sensor. We would like to compute

$$\hat{\theta} = \arg\min_{\theta \in \Theta} f(\theta) \tag{5.5}$$

where θ is a set of parameters which describe the global phenomena being sensed by the network and $f(\theta)$ is the cost function as defined in (5.3). The functions, $f_i : \mathbb{R}^d \to \mathbb{R}$ are convex (but not necessarily differentiable) and Θ is a nonempty, closed, and convex subset of $\mathbb{R}^d$.

The optimization problems can be solved iteratively by using gradient and subgradient methods. The update equation for a centralized subgradient descent approach to solve (5.5) is

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \alpha \sum_{i=1}^{N} g_{i,k} \tag{5.6}$$

where $g_{i,k} \in \partial f_i(\hat{\theta}^{(k)}), \alpha$ is a positive step size, and k is the iteration number. In this approach each update step uses data from all the sensors.

In a decentralized incremental approach, each iteration (5.6) is divided into N subiterations. In nth subiteration, nth sensor node updates its local parameter estimate$f_n(\theta)$. The algorithm can be written as:

$$\psi_0^{(k)} = \hat{\theta}^{(k-1)} \tag{5.7}$$

$$\psi_i^{(k)} = \psi_{i-1}^{(k)} - \alpha g_{i,k}, \quad i = 1, 2, \ldots, N \tag{5.8}$$

$$\hat{\theta}^{(k)} = \psi_N^{(k)} \tag{5.9}$$

where $\hat{\theta}^{(k)}$ is the estimated parameter vector obtained after k iterations and $\psi_N^{(k)}$ is the parameter estimate of Nth node in kth iteration. For analyzing the rate of convergence an arbitrary starting point is assumed.

The energy savings ratio between the use of an incremental optimization algorithm and a centralized optimization algorithm is shown to be [18]

$$R = c_3 M N^{1/d} \epsilon^2 \tag{5.10}$$

For N nodes with M readings each, a maximum estimation error ϵ, d the number of dimensions the sensor network is deployed in, and c_3 is the ratio between the number of bits required to describe the parameter vector θ and the measurements size in bits. Thus, as the number of readings or nodes in the network increases, it will become more advantageous to use an incremental algorithm for processing.

5.3 Robust Estimation

The main challenges in estimating parameters in a wireless sensor network are (i) link failure. (ii) impulsive noise. A noise level that fluctuates over a range greater than 10 dB during observation is classified as impulsive. Here we propose an distributed algorithm which is robust to link failure as well as impulsive noise while maintaining faster convergence and low residual mean square error(MSE). Bang et al[19] have proposed a proportional sign algorithm which is robust in the presence of contaminated-Gaussian noise, but this algorithm only involves a fixed nonlinear function [20]. Delouille, Neelamani and Baraniuk in [21] have proposed a method that minimizes the mean square error by using an iterative algorithm. Transmission of all data to a central processor and then estimation using techniques such as Wiener Filtering (with complexity $O(N^2)$), requires a large amount of communication. Instead the estimation process is divided among smaller groups of nodes which will interchange their measurements to give an optimal set of estimates. In this paper the robust estimation in incremental approach is shown using a real time estimation problem in sensor network. Suppose that a sensor network has been deployed over a region to find the average temperature. Each sensor collects a set of M temperature measurements, $\{x_{i,j}\}_{i=1}^{m}, j = 1, 2, \ldots, N$ over some period. At the end of the day the mean temperature

$$\hat{p} = \frac{1}{MN} \sum_{i,j} x_{i,j}$$

is to be calculated. Let us assume that the measurements are i.i.d. and the variance of each measurement is σ^2. However, some fraction say 10% of sensors are damaged or miscalibrated, so that they give reading with variance $100\sigma^2$. Then the estimator variance will increases by a factor of 10. Ideally, these bad measurements should be identified and discarded from the estimation process. Robust estimation techniques attempt to do so by modifying the cost function.

In a general estimation problem, the classical least-square loss function, $\|x - \theta\|^2$, is used. For robust estimation, the classical least-square loss function is replaced with a different robust function, $h(x, \theta)$. Typically the the robust function $h(x, \theta)$ is chosen to give less weight to data points which deviate greatly from the parameter, θ. So the cost

function (5.2) is modified for a robust estimation and given as

$$f_{robust}(\theta) = \frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M} h(x_{i,j},\theta)$$

Several robust functions are available in literature. The l_1 distance is one example of a robust function. Another standard robust cost function is the Huber loss function [18] as

$$h(x;\theta) = \begin{cases} \|x-\theta\|^2/2, & \text{for } \|x-\theta\| \leq \gamma \\ \gamma\|x-\theta\| - \gamma^2/2, & \text{for } \|x-\theta\| > \gamma \end{cases} \tag{5.11}$$

This function acts as usual squared error loss function if the distance between the data point x and γ is within a threshold value γ that means if x close to θ, but gives less weight to points outside a radius γ from the location θ.

Here another function known as error saturation non-linearity [22, 23] is proposed which is robust against link failure and Gaussian-contaminated impulsive noise. The cost function for error $e = \|x-\theta\|$ is defined as

$$h(e) = \int_0^e \exp[-u^2/2\sigma_s^2]du = \sqrt{\frac{\pi}{2}}erf\left[\frac{e}{\sqrt{2}\sigma_s}\right] \tag{5.12}$$

where σ_s is a parameter that defines the degree of saturation.

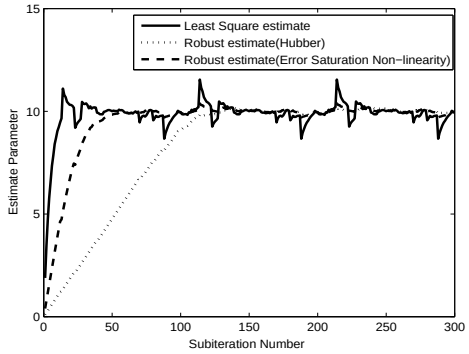
A distributed robust estimation algorithm is easily attained in the incremental sub-gradient framework by equating

$$f_i(\theta) = \frac{1}{M}\sum_{j=1}^{M} h(x_{i,j};\theta) \tag{5.13}$$

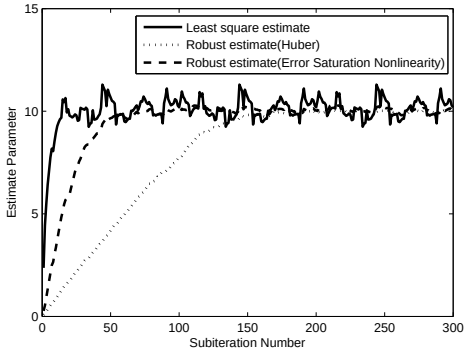
5.3.1 Robust incremental estimation during node failure

To show the robustness of different functions, the simulation work is carried out using a network where sensors are uniformly distributed over a homogeneous region, measurements are i.i.d. and corrupted by additive white Gaussian noise. In this example 100 sensor nodes are considered with each node collecting 10 measurements. However some sensors are damaged and give noisy measurements. Let $N(\mu,\sigma^2)$ denote the Gaussian distribution with mean μ and variance σ^2. A sensor which is working collects data with distribution $x_{i,j} \sim N(10,1)$, and a damaged sensor collects data with distribution $x_{i,j} \sim N(10,100)$. We have used our proposed error saturation non-linearity function

with $\sigma_s^2 = 10$ and compared its performance with the Hubber loss function.

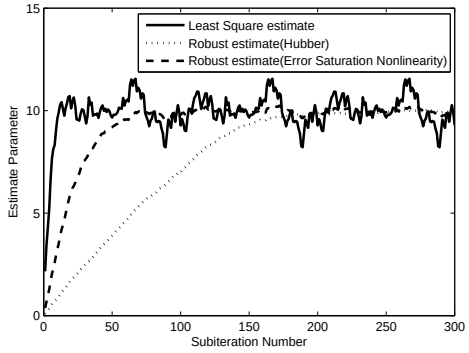


(a) 10% of the sensors are damaged

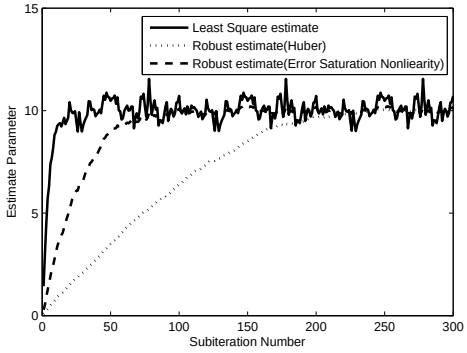


(b) 50% of the sensors are damaged

Figure 5.1: Robust incremental estimation procedures during node failure



(a) 10% of the sensors are damaged



(b) 50% of the sensors are damaged

Figure 5.2: Robust incremental estimation procedures during node failure and impulsive noise condition

The convergence characteristics of least-square estimate and the incremental robust estimate for both the robust functions are shown in Fig. 5.1(a), when 10% of the sensors are being damaged where as Fig. 5.1(b) depicts the convergence behavior when 50% nodes are damaged. Notice that the least square estimate converges faster than the robust estimate, but the variance of the distributed estimator is large. The results show that the robust estiamte obtained using saturation non-linearity converges faster as compared to that of Huber loss function.

5.3.2 Robust incremental estimation during node failure and impulsive noise condition

A contaminated-Gaussian impulsive noise (a two component Gaussian mixture)[24, 25, 26] is modeled as

$$v(i) = v_g(i) + v_{im}(i) = v_g(i) + b(i)v_w(i) \tag{5.14}$$

where $v_g(i)$ and $v_w(i)$ are independent zero mean Gaussian noise sequences with variances σ_g^2 and σ_w^2, respectively; $b(i)$ is a switch sequence of ones and zeros, which is modeled as an i.i.d. Bernoulli random process with probability of occurence $P_r(b(i) = 1) = p_r$ and $P_r(b(i) = 0) = 1 - p_r$. The variance of $v_w(i)$ is chosen to be much larger than that of $v_g(i)$ so that with $b(i) = 1$, a large impulse is experienced in $v(i)$. The corresponding pdf of $v(i)$ is given as

$$f_v(x) = \frac{1-p_r}{\sqrt{2\pi}\sigma_g} \exp\left(\frac{-x^2}{2\sigma_g^2}\right) + \frac{p_r}{\sqrt{2\pi}\sigma_\Sigma} \exp\left(\frac{-x^2}{2\sigma_\Sigma^2}\right) \tag{5.15}$$

where $\sigma_\Sigma^2 = \sigma_g^2 + \sigma_w^2$ and $E[v^2(i)] = \sigma_v^2 = \sigma_g^2 + p_r\sigma_w^2$.

The performance of the robust estimation algorithms in presence of impulsive noise is depicted in Figs. 5.2(a) and 5.2(b). The parameters of the algorithm are taken as follows: the step size $\alpha = 0.1$ and $\sigma_s^2 = 10, \sigma_g^2 = 10^{-3}, \sigma_w^2 = 10^4\sigma_g^2$. The results shows that the incremental robust estimation algorithms are robust to impulsive noise. The simulation results also reveal that the incremental robust estimate using error saturation non-linearity converges faster compared to Hubber loss function.

5.4 Conclusion

This paper has investigated a simple distributed algorithm for data processing in a wireless sensor network. The basic operation involves circulation of a parameter estimate through the network, and small adjustments to the estimate at each node based on its local data. These distributed algorithms can be viewed as incremental subgradient optimization procedure. A new cost function is proposed which is robust to node failure and impulsive noise. The simulation results show that the robust incremental estimate obtained using error saturation non-linearity is better than the estimate obtained by Huber loss function based method.

Chapter 6

Robust Distributed Least Mean Square Algorithm

6.1 Introduction

When data is contaminated with non-Gaussian noise, the conventional adaptive filters using mean square error criterion provides poor performance. In many physical environments the additive noise is modeled as impulsive and is characterized by long-tailed non-Gaussian distribution. The performance of the system is evaluated under the assumption that the Gaussian noise is severally degraded by the non-Gaussian or Gaussian mixture [24] noise due to deviation from normality in the tails [26, 25]. Nonlinear techniques are employed to reduce the effect of impulsive interference on the systems. The effects of saturation type of non-linearity on the least-mean square adaptation for Gaussian inputs and Gaussian noise have been studied [22, 27]. Recent research focus is to develop adaptive algorithm that are robust to impulsive noise or outliers present in the training data. Number of algorithms have been proposed [25, 28, 29, 30] to reduce the effects of impulsive noise. For example, in the order statistic leas mean-square LMS algorithm median filter is applied to reduce the adverse effects of impulsive noise[31]. Similarly, in the adaptive threshold nonlinear [20], nonlinear clipping function is used to limit the transient fluctuation. This class of algorithms is difficult to analyze and therefore it is not uncommon to resort to different methods and assumptions. Prof Neil J. Bershad [23] has shown that LMS with error saturation nonlineary provides good performance in presence of impulsive noise.

The error nonlinearity analysis [17, 32] and data nonlinearity analysis [16] has been done using weighted-energy conservation method. The theory dealt in [23] provides the idea for subsequent analysis in presence of Gaussian mixture. It also suggests that how it can be applied to each component separately to obtain recursive relation for the nonlinear LMS.

In this chapter we use both the ideas to develop a new generalized method to obtain the network that should be robust to impulsive noise. The steady-state analysis of saturation nonlinearity incremental LMS in presence of contaminated Gaussian impulsive noise is done here and its robustness over the conventional incremental LMS algorithm is shown both theoretically and using simulation. This analysis can be extended for distributed diffusion LMS algorithm. For simplicity the assumptions and notations used in chapter 2 are also followed in this chapter.

6.2 Distributed Incremental LMS Algorithm with Error Saturation

The concept of incremental algorithms was discussed in chapter 2. But the LMS based incremental algorithm is not robust to impulsive noise. So here we introduce a new class of LMS algorithm based on error saturation nonlinearity.

6.2.1 Adaptive Algorithm with Error Saturation Nonlinearity

The weight update equation for LMS algorithm is given as

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu e(i)\mathbf{u}_i^T$$

In this pchapter we focus on a different class of algorithms by introducing an error nonlinearity term into the feedback error signal so that the weight update equation can be written as

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu \mathbf{u}_i^T f[e(i)] \tag{6.1}$$

where

$$f(y) = \int_0^y \exp[-u^2/2\sigma_s^2]du = \sqrt{\frac{\pi}{2}} erf\left[\frac{y}{\sqrt{2}\sigma_s}\right] \tag{6.2}$$

Here σ_s is a parameter that defines the degree of saturation, $\mathbf{w}_i$ is the estimate of $\mathbf{w}$ at time i and μ is the step size

$$e(i) = d(i) - \mathbf{u}_i\mathbf{w}_{i-1} = \mathbf{u}_i\mathbf{w}^\circ - \mathbf{u}_i\mathbf{w}_{i-1} + v(i) \tag{6.3}$$

If $f(e)$ represents the cost function, then its gradient is defined as

$$\begin{aligned}\frac{\partial f}{\partial \mathbf{w}} &= \frac{\partial f}{\partial e} \cdot \frac{\partial e}{\partial \mathbf{w}} \\ &= \frac{\partial f}{\partial e} \cdot (-\mathbf{u})\end{aligned}$$

If we choose the cost function $f(e) = e^2$ then $\frac{\partial f}{\partial e} = 2e$ is linear. For higher order of e, $\frac{\partial f}{\partial e}$ is a non-linear function of e. In this approach nonlinear $f(e)$ and is so chosen that of $\frac{\partial f}{\partial e}$ is also nonlinear *i. e.*.

$$f(e) = \int \left(\frac{\partial f}{\partial e}\right) de$$

In this saturation non-linearity LMS case we have chosen Gaussian nonlinearity on error.

6.2.2 Model for Impulsive Noise

A contaminated-Gaussian impulsive noise (a two component Gaussian mixture)[24, 25, 26] is modeled as

$$v(i) = v_g(i) + v_{im}(i) = v_g(i) + b(i)v_w(i) \tag{6.4}$$

where $v_g(i)$ and $v_w(i)$ are independent zero mean Gaussian noise sequences with variances σ_g^2 and σ_w^2, respectively; $b(i)$ is a switch sequence of ones and zeros, which is modeled as an i.i.d. Bernoulli random process with probability of occurence $P_r(b(i) = 1) = p_r$ and $P_r(b(i) = 0) = 1 - p_r$. The variance of $v_w(i)$ is chosen to be much larger than that of $v_g(i)$ so that with $b(i) = 1$, a large impulse is experienced in $v(i)$. The corresponding pdf of $v(i)$ is given as

$$f_v(x) = \frac{1 - p_r}{\sqrt{2\pi}\sigma_g} \exp\left(\frac{-x^2}{2\sigma_g^2}\right) + \frac{p_r}{\sqrt{2\pi}\sigma_{tot}} \exp\left(\frac{-x^2}{2\sigma_{tot}^2}\right) \tag{6.5}$$

where $\sigma_{tot}^2 = \sigma_g^2 + \sigma_w^2$ and $E[v^2(i)] = \sigma_v^2 = \sigma_g^2 + p_r\sigma_w^2$. Note that when $p_r = 0$ or 1, $n_o(i)$ is a zero-mean Gaussian variate. Otherwise $n_o(i)$ is non-gaussian.

6.2.3 Incremental LMS Algorithm with Error Saturation Non-linearity

The distributed algorithm defined in (2.1) is very simple and it gives good performance in stationary environment. But sensor network is used in an environment where impulsive noise is present. So the distributed incremental LMS algorithm is modified by adding non-linearity in the error term wich is defined in (6.3). The algorithm is defined as follows For each time $i \geq 0$, repeat:

$$\begin{aligned} k &= 1, \ldots, N \\ \Psi_0^{(i)} &= \mathbf{w}_{i-1} \\ e_k(i) &= d_k(i) - \mathbf{u}_{k,i}\Psi_{k-1}^{(i)} \\ \Psi_k^{(i)} &= \Psi_{k-1}^{(i)} + \mu_k \mathbf{u}_{k,i}^T f(e_k(i)) \\ \mathbf{w}_i &= \Psi_N^{(i)} \end{aligned} \tag{6.6}$$

This cooperative scheme is defined in [4] which gives better performance than nondistributed algorithm. Here we can improve the steady-state performance of incremental LMS in presence of Gaussian noise by choosing suitable value for degree of saturation. This algorithm is more complex than incremental LMS as it calculates the error nonlinearity term.

6.3 Performance Analysis

The performance analysis of incremental LMS which was discussed in chapter 2 is used here.

6.3.1 Data Models and Assumptions

For simplicity, the data model used in (2.2) is used and the assumptions used in chapter 2 are followed. Now the saturation nonlinearity incremental LMS which is defined in (6.6) can be written as

$$\Psi_k^{(i)} = \Psi_{k-1}^{(i)} + \mu_k \mathbf{u}_{k,i}^{\mathrm{T}} f(e_k(i))$$

Subtracting $\mathbf{w}^{\circ}$ from both sides of above equation, we obtain

$$\tilde{\Psi}_k^{(i)} = \tilde{\Psi}_{k-1}^{(i)} - \mu_k \mathbf{u}_{k,i}^{\mathrm{T}} f(e_k(i)) \tag{6.7}$$

Relation between various error terms $e_{a,k}^{\Sigma}(i)$, $e_{p,k}^{\Sigma}(i)$ and $e_k(i)$ is obtained by premultiplying both sides of (6.7) by $\mathbf{u}_{k,i}\Sigma$ gives

$$\begin{aligned} e_{p,k}^{\Sigma}(i) &= e_{a,k}^{\Sigma}(i) - \mu_k \|\mathbf{u}_{k,i}\|_{\Sigma}^2 f[e_k(i)] \\ f[e_k(i)] &= \frac{1}{\mu_k} \frac{e_{a,k}^{\Sigma}(i) - e_{p,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \end{aligned} \tag{6.8}$$

6.3.2 Weight-Energy Relation

Eliminating the term $f[e(i)]$ from (6.7) by using (6.8), we obtain

$$\tilde{\Psi}_k^{(i)} + \frac{\mathbf{u}_{k,i} e_{a,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} = \tilde{\Psi}_{k-1}^{(i)} + \frac{\mathbf{u}_{k,i} e_{p,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \tag{6.9}$$

Taking weighted norm of both sides of (6.9), we obtain

$$\|\tilde{\Psi}_k^{(i)}\|_\Sigma^2 + \frac{|e_{a,k}^\Sigma(i)|^2}{\|\mathbf{u}_{k,i}\|_\Sigma^2} = \|\tilde{\Psi}_{k-1}^{(i)}\|_\Sigma^2 + \frac{|e_{p,k}^\Sigma(i)|^2}{\|\mathbf{u}_{k,i}\|_\Sigma^2} \tag{6.10}$$

The above equation is known as space-time version of the weighted energy conservation relation.

6.3.3 Variance Relation

Substiuting (6.8) into (6.10) and taking expectation on both sides leads to

$$E\|\tilde{\Psi}_k^{(i)}\|_\Sigma^2 = E\|\tilde{\Psi}_{k-1}^{(i)}\|_\Sigma^2 - 2\mu_k E e_{a,k}^\Sigma(i) f[e_k(i)] + \mu_k^2 E\|\mathbf{u}_{k,i}\|_\Sigma^2 f^2[e_k(i)] \tag{6.11}$$

Evaluation of 2nd and 3rd term on RHS of (6.11) is difficult as it contains the nonlinearity term. To evaluate the transient analysis we make the same assumption as taken in [17].

- The noise sequence $v_k(i)$ is independence of $\mathbf{u}_{k,i}$
- For any constant matrix Σ and for all i, $e_{a,k}(i)$ and $e_{a,k}^\Sigma(i)$ are jointly Gaussian.
- The adaptive filter is long enough such that the weighted norm of input regressor and the square of error nonlinearity i.e. $f^2[e_k(i)]$ are uncorrelated.

The Price's theorem [33, 34] plays an important rule in analyzing the 2nd term on RHS of equation(6.11) and is given as

$$E[af[b+c]] = \frac{E[ab]}{E[b^2]} E[bf[b+c]]$$

where a and b are jointly Gaussian random variables that are independent from the third random variable c. In [23], [17], [4] the noise is considered as simply Gaussian and independent of the errors $e_{a,k}(i)$ and $e_{a,k}^\Sigma(i)$. In this chapter we consider that the noise is a contaminated-Gaussian impulsive noise and is independent of the errors. Using Price's theorem we get the same equation as in [17],

$$E[e_{a,k}^\Sigma(i) f[e_k(i)]] = E[e_{a,k}^\Sigma(i) e_{a,k}(i)] h_{G,k} E[e_{a,k}^2(i)] \tag{6.12}$$

where $h_{G,k}$ contains the non-linear term and is given as

$$h_{G,k}Ee_{a,k}^2(i) = \frac{E[e_{a,k}(i)f[e_{a,k}(i)+v_k(i)]]}{E[e_{a,k}^2(i)]} \tag{6.13}$$

The general expression for $h_{G,k}$ for any type of noise which is given in [17] is

$$h_{G,k} = \frac{\sigma_s}{\sqrt{E[e_{a,k}^2(i)]+\sigma_s^2}} E\left[\exp\left[-\frac{v_k^2(i)}{2(E[e_{a,k}^2(i)]+\sigma_s^2)}\right]\right]$$

The polarization property of weighted norm says that

$$(\mathbf{u}\Sigma_1\mathbf{w})(\mathbf{u}\Sigma_2\mathbf{w}) = \|\mathbf{w}\|^2_{\Sigma_1\mathbf{u}^T\mathbf{u}\Sigma_2}$$

Using this property we can write

$$E[e_{a,k}^{\Sigma}(i)e_{a,k}(i)] = E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{\Sigma\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} \tag{6.14}$$

Then (6.12) can be written as

$$E[e_{a,k}^{\Sigma}(i)f[e_k(i)]] = E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{\Sigma\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} h_{G,k}E[e_{a,k}^2(i)] \tag{6.15}$$

In a similar way we can evaluate the third term of (6.11) by taking the assumption of a long filter for which the weighted norm of input data and the squared error nonlinearity are uncorrelated as in [17]. So the third term can be written as

$$\begin{aligned} E[\|\mathbf{u}_{k,i}\|_{\Sigma}^2 f^2[e_k(i)]] &= E\|\mathbf{u}_{k,i}\|_{\Sigma}^2 Ef^2[e_k(i)] \\ &= E\|\mathbf{u}_{k,i}\|_{\Sigma}^2 h_{U,k}E[e_{a,k}^2(i)] \end{aligned} \tag{6.16}$$

where the nonlinear component $h_{U,k}$ is given by

$$h_{U,k}E[e_{a,k}^2(i)] = Ef^2[e_k(i)] \tag{6.17}$$

The general expression of $h_{U,k}$ for any type of noise as given in [17] is

$$h_{U,k} = 2\pi\sigma_s^2\left(\frac{1}{4} - \frac{1}{\pi}\int_{\pi/4}^{\pi/2}\sqrt{\frac{\sigma_s^2\sin^2(\theta)}{E[e_{a,k}^2]+\sigma_s^2\sin^2(\theta)}}\cdot E\left[\exp\left[-\frac{v_k^2(i)}{2(E[e_{a,k}^2]+\sigma_s^2\sin^2(\theta))}\right]\right]\right) \tag{6.18}$$

After evaluating the 2nd and 3rd term using few realistic assumptions, the variance relation is obtained as

$$E\|\tilde{\Psi}_k^{(i)}\|_\Sigma^2 = E\|\tilde{\Psi}_{k-1}^{(i)}\|_\Sigma^2 - 2\mu_k E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{\Sigma\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} h_{G,k} E e_{a,k}^2(i) + \mu_k^2 E\|\mathbf{u}_{k,i}\|_\Sigma^2 h_{U,k} E e_{a,k}^2(i) \tag{6.19}$$

6.3.4 Evaluation of nonlinear term h_G and h_U

If the noise is Gaussian, then we can use the expressions for h_G and h_U by puting the Gaussian probability density function into the general expression which are given in [4]. Here the noise is contaminated-Gaussian impulsive whose probability density function is defined in (6.5). So here we evaluate the expressions for h_G and h_U from their general definition by puting the new probability density function defined in (6.5). Therefore in presence of impulsive noise the expressions for $h_{G,k}$ and $h_{U,k}$ are given as,

$$h_{G,k} = \frac{(1-p_r)\sigma_s}{\sqrt{E[e_{a,k}^2] + \sigma_{g,k}^2 + \sigma_s^2}} + \frac{p_r\sigma_s}{\sqrt{E[e_{a,k}^2] + \sigma_{tot}^2 + \sigma_s^2}} \tag{6.20a}$$

$$h_{U,k} = (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{g,k}^2 + E[e_{a,k}^2]}{E[e_{a,k}^2] + \sigma_s^2 + \sigma_{g,k}^2}\right) + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{tot}^2 + E[e_{a,k}^2]}{E[e_{a,k}^2] + \sigma_s^2 + \sigma_{tot}^2}\right) \tag{6.20b}$$

Calling upon the independence of the regression data $\{\mathbf{u}_k\}$, it allows us to write

$$E[e_{a,k}^2(i)] = E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} = E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}} \tag{6.21}$$

which is the excess-mean-square error of node k in the network. Now $h_{G,k}$ and $h_{U,k}$ are expressed as

$$h_{G,k}^{(i)} = \frac{(1-p_r)\sigma_s}{\sqrt{E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_{g,k}^2 + \sigma_s^2}} + \frac{p_r\sigma_s}{\sqrt{E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_{tot}^2 + \sigma_s^2}} \tag{6.22a}$$

$$h_{U,k}^{(i)} = (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{g,k}^2 + E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}}}{E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_s^2 + \sigma_{g,k}^2}\right) + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{tot}^2 + E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}}}{E\|\tilde{\Psi}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_s^2 + \sigma_{tot}^2}\right) \tag{6.22b}$$

6.3.5 Gaussian Regressor

For simple analysis the Gaussian regressor discussed in section 2.3.4 is taken into account. Under the change of variables defined in section 2.3.4, the variance relation (6.19) is written as

$$E\|\overline{\Psi}_k^{(i)}\|_{\overline{\Sigma}}^2 = E\|\overline{\Psi}_{k-1}^{(i)}\|_{\overline{\Sigma}}^2 - 2\mu_k E\|\overline{\Psi}_{k-1}^{(i)}\|_{\overline{\Sigma}\overline{\mathbf{u}}_{k,i}\overline{\mathbf{u}}_{k,i}^T}^2 h_{G,k}^{(i)} + \mu_k^2 E\|\overline{\mathbf{u}}_{k,i}\|_{\overline{\Sigma}}^2 h_{U,k}^{(i)} \tag{6.23}$$

Using equation(2.29) and invoking the independence of the regression data $\{\mathbf{u}_k\}$, (6.23) can be written as

$$E\|\overline{\Psi}_k^{(i)}\|_{\overline{\Sigma}}^2 = E\|\overline{\Psi}_{k-1}^{(i)}\|_{\overline{\Sigma}}^2 - 2\mu_k E\|\overline{\Psi}_{k-1}^{(i)}\|_{\overline{\Sigma}\wedge_k}^2 h_{G,k}^{(i)} + \mu_k^2 \mathrm{Tr}(\wedge_k\overline{\Sigma}) h_{U,k}^{(i)} \tag{6.24}$$

6.3.6 Steady-State Behavior

When $i \to \infty$, let us take $\overline{\mathbf{h}}_k = \overline{\Psi}_k^{\infty}$. Then for $i \to \infty$(i.e. in steady state), the variance relation (6.24) gives

$$E\|\overline{\mathbf{h}}_k\|_{\overline{\Sigma}}^2 = E\|\overline{\mathbf{h}}_{k-1}\|_{\overline{\Sigma}}^2 - 2\mu_k E\|\overline{\mathbf{h}}_{k-1}\|_{\overline{\Sigma}\wedge_k}^2 h_{G,k} + \mu_k^2 \mathrm{Tr}(\wedge_k\overline{\Sigma}) h_{U,k} \tag{6.25}$$

The convergence analysis of the LMS algorithm with a general error nonlinearity and an IID input were studied in [35, 32]. The theory says that the saturation LMS algorithm converge in its mean and variance. The performance of this algorithm was also studied [23], where the analysis proves its robustness to impulsive noise. Assuming that the variance is converging to the minimum value, let us approximate the nonlinear parameters in the above variance relation.

$$\begin{aligned} h_{G,k}^{(\infty)} &= \frac{(1-p_r)\sigma_s}{\sqrt{E\|\overline{\mathbf{h}}_{k-1}\|_{\wedge_k}^2 + \sigma_{g,k}^2 + \sigma_s^2}} + \frac{p_r\sigma_s}{\sqrt{E\|\overline{\mathbf{h}}_{k-1}\|_{\wedge_k}^2 + \sigma_{tot}^2 + \sigma_s^2}} \\ &\approx \frac{(1-p_r)\sigma_s}{\sqrt{\sigma_{g,k}^2 + \sigma_s^2}} + \frac{p_r\sigma_s}{\sqrt{\sigma_{tot}^2 + \sigma_s^2}} \end{aligned} \tag{6.26}$$

In similar way the other nonlinear parameter $h_{U,k}$ is given as

$$\begin{aligned} h_{U,k}^{(\infty)} &= (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{g,k}^2 + E\|\overline{\mathbf{h}}_{k-1}\|_{\wedge_k}^2}{E\|\overline{\mathbf{h}}_{k-1}\|_{\wedge_k}^2 + \sigma_s^2 + \sigma_{g,k}^2}\right) \\ &\quad + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{tot}^2 + E\|\overline{\mathbf{h}}_{k-1}\|_{\wedge_k}^2}{E\|\overline{\mathbf{h}}_{k-1}\|_{\wedge_k}^2 + \sigma_s^2 + \sigma_{tot}^2}\right) \\ &\approx (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{g,k}^2}{\sigma_s^2 + \sigma_{g,k}^2}\right) + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{tot}^2}{\sigma_s^2 + \sigma_{tot}^2}\right) \end{aligned} \tag{6.27}$$

Since the above parameters are independent of weighted matrix σ, it can be taken as a constant in further analysis. Therefore we can write (6.25) as

$$E\|\overline{\mathbf{h}}_k\|_{\overline{\Sigma}}^2 = E\|\overline{\mathbf{h}}_{k-1}\|_{\overline{\Sigma}}^2 - E\|\overline{\mathbf{h}}_{k-1}\|_{2\mu_k h_{G,k}\overline{\Sigma}\wedge_k}^2 + \mu_k^2 \mathrm{Tr}(\wedge_k\overline{\Sigma})h_{U,k} \tag{6.28}$$

The superposition property of *weighted-norm* says that

$$a_1\|x\|_{W_1}^2 + a_2\|x\|_{W_2}^2 = \|x\|_{a_1W_1+a_2W_2}^2$$

Using above property, (6.28) can be rewritten more compactly as

$$E\|\overline{\mathbf{h}}_k\|_{\overline{\Sigma}}^2 = E\|\overline{\mathbf{h}}_{k-1}\|_{\overline{\Sigma}'}^2 + \mu_k^2 \mathrm{Tr}(\wedge_k\overline{\Sigma})h_{U,k} \tag{6.29}$$

where

$$\overline{\Sigma}' = \overline{\Sigma} - 2\mu_k h_{G,k}\overline{\Sigma}\wedge_k \tag{6.30}$$

Using the vectors $\{\overline{\sigma}, \lambda_k\}$ defined in (2.33) of section 2.3.5, the expression (6.30) can be rewritten as

$$\begin{aligned} \overline{\sigma}' &= (\mathrm{I} - 2\mu_k h_{G,k}\wedge_k)\overline{\sigma} \\ &= \overline{F}_k\overline{\sigma} \end{aligned} \tag{6.31}$$

where the $M \times M$ coefficient matrix $\overline{F}_k$ is defined by

$$\overline{F}_k = \mathrm{I} - 2\mu_k h_{G,k}\wedge_k \tag{6.32}$$

We can rewrite (6.29) by using the vectors $\{\overline{\sigma}, \overline{\sigma}', \lambda_k\}$ instead of the matrices $\{\overline{\Sigma}, \overline{\Sigma}', \lambda_k\}$. Using (6.32) and the notation (2.33), (6.29) becomes

$$E\|\overline{\mathbf{h}}_k\|^2_{\mathrm{diag}\{\overline{\sigma}\}} = E\|\overline{\mathbf{h}}_{k-1}\|^2_{\mathrm{diag}\{\overline{F}_k\overline{\sigma}\}} + \mu_k^2 \mathrm{Tr}(\wedge_k \overline{\Sigma}) h_{U,k} \tag{6.33}$$

If we use the fact that $\mathrm{Tr}(\wedge_k \overline{\Sigma}) = \lambda_k^T \overline{\sigma}$ then (6.33) can be written as

$$E\|\overline{\mathbf{h}}_k\|^2_{\mathrm{diag}\{\overline{\sigma}\}} = E\|\overline{\mathbf{h}}_{k-1}\|^2_{\mathrm{diag}\{\overline{F}_k\overline{\sigma}\}} + \mu_k^2 (\lambda_k^T \overline{\sigma}) h_{U,k} \tag{6.34}$$

For the sake of compactness, the diag{} notation will be dropped from the subscripts, keeping only the corresponding vectors

$$E\|\overline{\mathbf{h}}_k\|^2_{\overline{\sigma}_k} = E\|\overline{\mathbf{h}}_{k-1}\|^2_{\overline{F}_k\overline{\sigma}_k} + \mu_k^2 (\lambda_k^T \overline{\sigma}) h_{U,k} \tag{6.35}$$

Let us take $\mathbf{g}_k = \mu_k^2 h_{U,k} \lambda_k^T$ as a row vector. Continuing the derivation as done in section 2.3.6, the expressions of MSD, EMSE and MSE are obtained which are similar to (2.45), (2.46) and (2.47). But the basic parameter $\overline{F}_k$ defined in (6.32) and $\mathbf{g}_k = \mu_k^2 h_{U,k} \lambda_k^T$ are totally different.

Similar to section 2.3.6 Φ can be approximated as

$$\begin{aligned}\Phi &= (\mathrm{I} - 2\mu_1 h_{G,1} \wedge_1)(\mathrm{I} - 2\mu_2 h_{G,2} \wedge_2) \ldots (\mathrm{I} - 2\mu_N h_{G,N} \wedge_N) \\ &\approx \mathrm{I} - (2\mu_1 h_{G,1} \wedge_1 + 2\mu_2 h_{G,2} \wedge_2 + \ldots + 2\mu_N h_{G,N} \wedge_N)\end{aligned}$$

so that

$$\mathrm{I} - \Phi \approx 2\mu_1 h_{G,1} \wedge_1 + 2\mu_2 h_{G,2} \wedge_2 + \ldots + 2\mu_N h_{G,N} \wedge_N$$

Now from (2.45), we can obtain the approximate relation for η_k as

$$\begin{aligned}\eta_k \approx\ & (\mu_1^2 h_{U,1} \lambda_1^T + \ldots + \mu_N^2 h_{U,N} \lambda_N^T) \times \\ & (2\mu_1 h_{G,1} \wedge_1 + 2\mu_2 h_{G,2} \wedge_2 + \ldots + 2\mu_N h_{G,N} \wedge_N)^{-1} q\end{aligned} \tag{6.36}$$

In a similar way, the EMSE can be approximated as

$$\zeta_k \approx (\mu_1^2 h_{U,1} \lambda_1^T + \ldots + \mu_N^2 h_{U,N} \lambda_N^T)(2\mu_1 h_{G,1} \wedge_1 + \ldots + 2\mu_N h_{G,N} \wedge_N)^{-1} \lambda_k \tag{6.37}$$

Both MSE and EMSE goes to zero asymptotically when the step size $\mu_l \to 0$, causing the MSE to achieve the background impulsive noise level everywhere.

6.4 Results and Discussion

In order to evaluate the performance of the error saturation nonlinearity incremental LMS algorithm, we provide the simulations comparing the theoretical performance to simulation results and also comparison with distributed incremental LMS. For comparison purpose the simulation parameters defined in section 2.4 are taken here. These parameters are depicted in Fig. 6.1 and Fig. 6.2. The background Gaussian noise variance $\sigma_{g,k}$ is shown in Fig. 6.3.

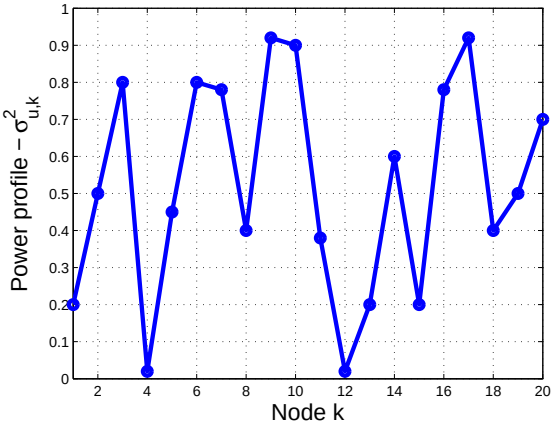


Figure 6.1: Regressor power profile

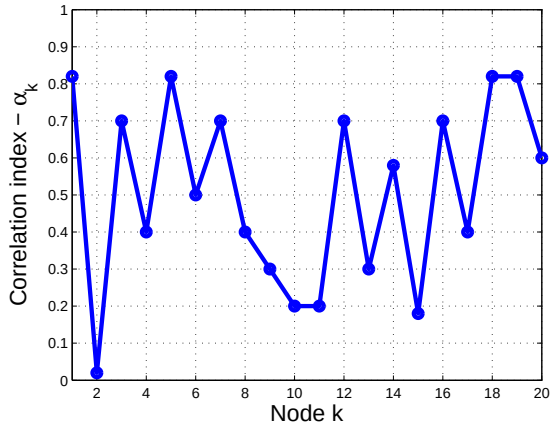


Figure 6.2: Correlation index per node

For all simulations μ was kept as 0.03 and σ_s^2 as 0.01. Initially we tested the robustness of the algorithm in 10% impulsive noise where the variance of Gaussian contaminated

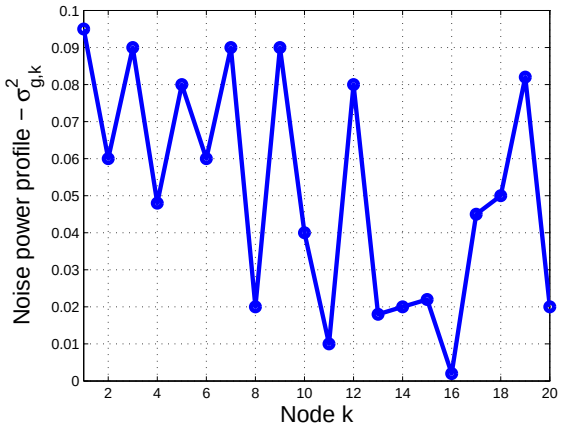


Figure 6.3: Noise power profile

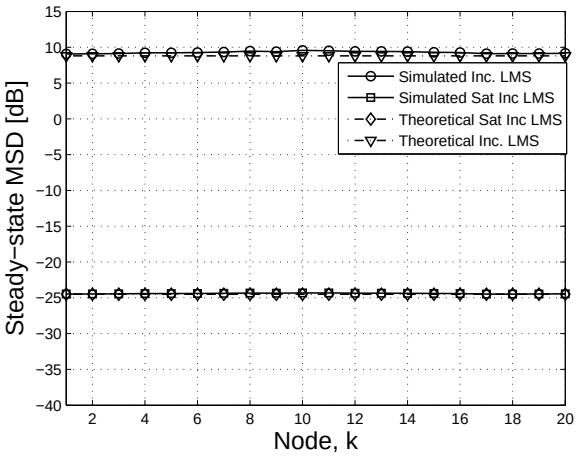


Figure 6.4: Theoretical and simulated MSD Vs nodes curve for $p_r = 0.1$

impulsive noise variance is defined as 10^4 times the back ground noise variance defined for individual node. The figures 6.4, 6.5 and 6.6 clearly shows the robustness of the algorithm over incremental LMS. The steady-state values of both the MSD and EMSE are approaching to good values around -25dB for MSD and -30dB for EMSE respectively where for the distributed incremental LMS the performances are 10dB and 5dB for MSD and EMSE respectively. But in both the cases the MSE are nearly same and shows that the mean-square error(MSE) does not converge.

The main objective of all the adaptive algorithms are to estimate weights that should approach to the optimum. When the noise is non stationary, then the MSE will not converge. The estimated weights using LMS algorithm will also diverge from the desired values because the error is used directly in the weight update equation of the adaptive

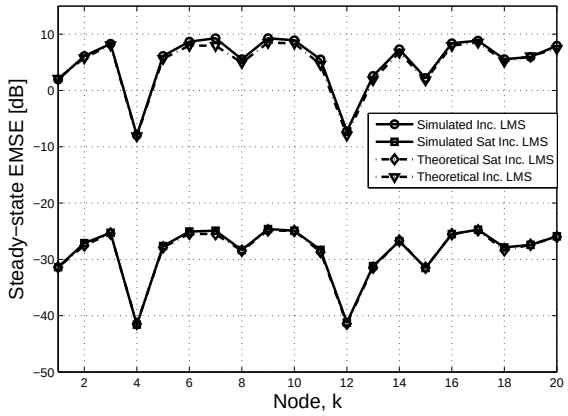


Figure 6.5: Theoretical and simulated EMSE Vs nodes curve for $p_r = 0.1$

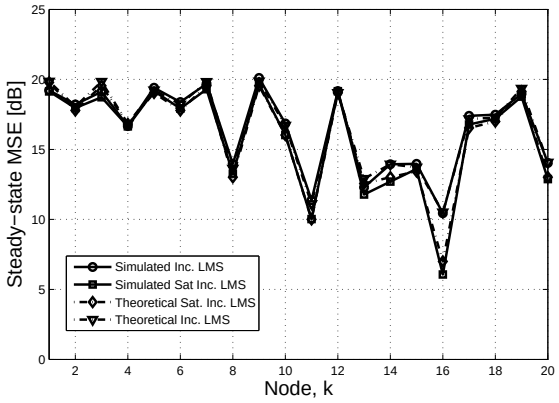


Figure 6.6: Theoretical and simulated EMSE Vs nodes curve for $p_r = 0.1$

algorithm. Therefore the MSD and EMSE are not converging and their steady-state values are very high in case of distributed incremental LMS. In the distributed incremental LMS algorithm with error saturation nonlinearity, the error is not used directly in the weight update equation. Here the error is fed back through a Gaussian nonlinear function, where the error is mapped within a limit of $[-\sqrt{\frac{\pi}{2}}\sigma_s, \sqrt{\frac{\pi}{2}}\sigma_s]$ depending on the value of σ_s. The error may be high enough due to impulsive noise, but is mapped to a small value with in the defined limit. So the estimated weights approach towards the desired weight due to the presence of error nonlinearity in the update equation. This reflects the steady-state performance of the filter. The MSD and EMSE are very low indicating Ψ_k^∞ is a good estimate for $\mathbf{w}^\circ$. But since the error remains unchanged, so the steady-state MSE does not converge in both the cases.

After showing the robustness of algorithm in 10% impulsive noise we tested it for 50%.

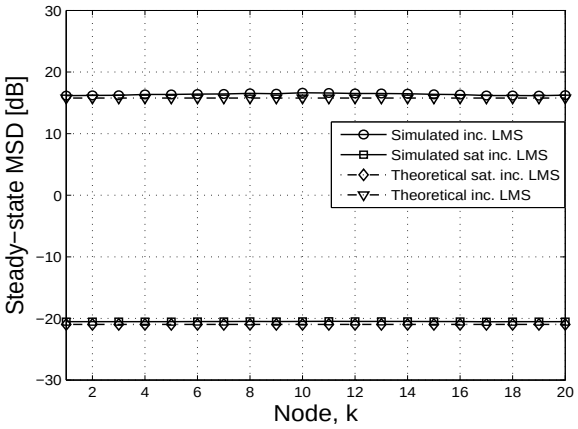


Figure 6.7: Theoretical and simulated MSD Vs nodes curve for $p_r = 0.5$

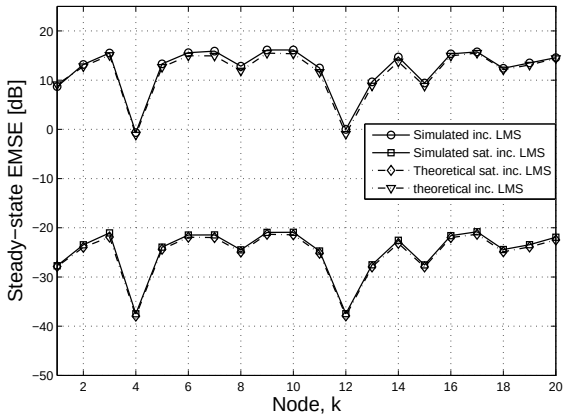


Figure 6.8: Theoretical and simulated EMSE Vs nodes curve for $p_r = 0.5$

The figures 6.7, 6.8 and 6.9 shows the error saturation nonlinearity incremental LMS algorithm's robustness towards the impulsive noise. When the probability of occurrence of impulsive noise is 50%, the estimated parameters also approach towards desired values.

A robust adaptive algorithm, together with a good step-size and proper cooperative scheme may take advantage of the spatial diversity provided by the adaptive network. A small step-size should be assigned to the nodes with poor performance.

6.5 Conclusion

In this chapter steady-state performance of distributed incremental LMS algorithm with error saturation nonlinearity was discussed which can also be extended to distributed

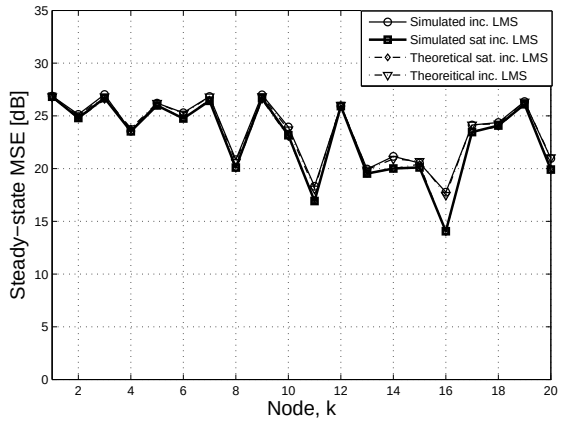


Figure 6.9: Theoretical and simulated MSE Vs nodes curve for $p_r = 0.5$

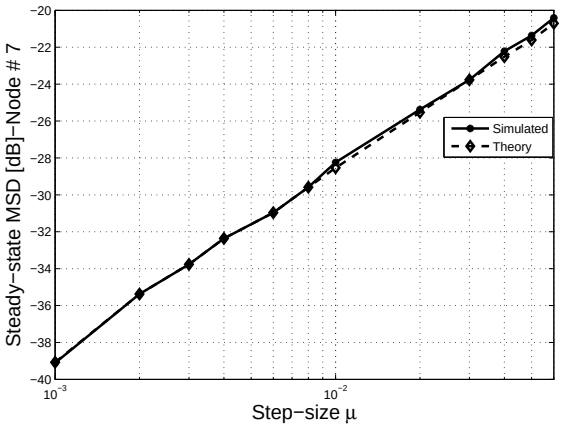


Figure 6.10: Theoretical and simulated MSD Vs step-size at node-7 for $p_r = 0.2$

diffusion LMS algorithm. It shows the robustness of proposed algorithm over the conventional incremental LMS both in theory and simulation. The steady-state expressions for MSE, EMSE and MSD have been derived and was found to be matching very well with simulation results.

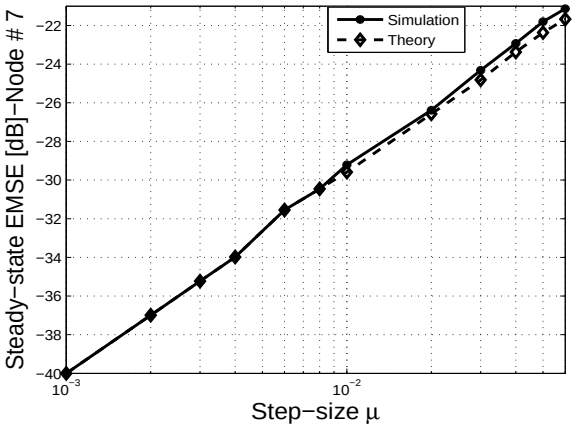


Figure 6.11: Theoretical and simulated EMSE Vs step-size at node-7 for $p_r = 0.2$

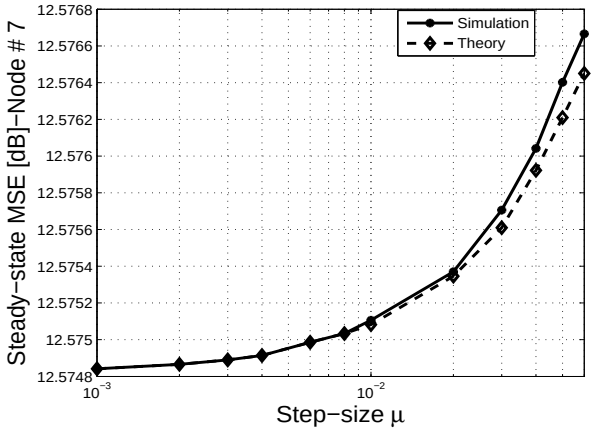


Figure 6.12: Theoretical and simulated MSE Vs step-size at node-7 for $p_r = 0.2$

Chapter 7

Energy Efficient Layout for Wireless Sensor Network

7.1 Introduction

The goal of the sensor placement problem is to determine optimal locations of sensors for maximizing the information collected from the sensor network. But equally important are factors such as the energy consumption of the network. This results in a tradeoff between multiple objectives such as coverage and energy consumption. In this chapter, we address the optimization of the multiple objectives described above by adopting a multi-objective optimization framework.

7.2 Related work

Several researchers have investigated the sensor placement problem focusing on detection and coverage. Two noteworthy contributions on this problem are by Dhillon et al. [36] and Clouqueur et al. [37]. Dhillon et al. [36] have considered the sensor placement problem for grid coverage. They have developed an iterative sensor placement algorithm for coverage optimization under the constraints of terrain properties. The sensor model for target detection used in [36] does not consider the physics of the phenomenon or the measurement noise. Clouqueur et al. [37] have solved the sensor placement problem for distributed target detection. They have developed a sequential deployment algorithm which terminates when a satisfactory detection performance is achieved. They have focused on deployment cost and detection performance while ignoring energy efficiency. Kar and Banerjee [38] solve the problem of deploying a minimal number of homogeneous sensors to cover a plane with a connected sensor network. The main goal of their approach is to minimize the number of sensors while guaranteeing coverage and connectivity. All of these works have considered the sensor deployment problem for single objective optimization in which the focus was on maximizing the detection performance or minimizing the deployment cost. In addition, energy efficiency has not been considered by several existing placement strategies. Ferentinos and Tsiligiridis [39] solve a multi-objective sensor network design problem in which sensors are selectively activated and a specific set of nodes are selected as cluster heads to monitor an agricultural area. The objectives are to optimize energy consumption and application requirements such as uniformity of the sensor measurements while incorporating constraints on the connectivity of the network. They combine the multiple objectives using weights that represent the relative priorities of various objectives, and use a genetic algorithm for solving the resulting single-objective optimization problem. Though their approach is interesting, it does not provide Pareto-

optimal solutions characterizing the tradeoffs between the objectives. In addition, the system designer has to determine the weights a priori which might be infeasible if the relative priorities of different objectives are unknown.

7.3 Problem formulation

7.3.1 WSN Modeling

It is assumed that each node knows its position in the search space and all sensor nodes are homogeneous. High energy communication node(HECN) is assumed to be more powerful than sensor nodes. A flat square surface is considered in which HECN is placed at the center for convenience. The sensing area of each node is assumed to have a circular shape with radius R_{sens}. The communication range of each node is defined by the area of a circle with radius R_{comm}. Initially nodes are assumed to have equal energy. It is assumed that for every data transmission, the energy decreases by one unit. The co-ordinates of the sensor nodes $(x_1, y_1), (x_2, y_2), \ldots\ldots$ are considered as design variables. Sensor nodes are assumed to have certain mobility.

7.3.2 Calculation of Objectives

The following two objectives are considered.

- Maximize the total coverage of the sensor network: f_1
- Maximize the lifetime of the sensor network: f_2

Detailed description of objective functions f_1 and f_2 are presented below.

Coverage

It is one of the measurements of QoS of a sensor network. The coverage of each sensor can be defined either by a binary sensor model [40] shown in Figure 7.1 or a stochastic sensor model [41] shown in Figure 7.2.

In the binary sensor model shown in Figure 7.1, the detection probability of the event of interest is one within the sensing range; otherwise, the probability is zero. Coverage of a network using binary sensor model is determined by finding the union of sensing areas defined by location of each sensor and Rsens. Although the binary sensor model is

simpler, it does not take the uncertainty factor in sensor measurement into consideration. The binary sensor model is given by

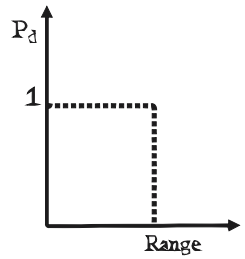


Figure 7.1: Binary sensor coverage model

$$C_{ij}(x,y) = \begin{cases} 1 & \text{for} \quad d_{ij}(x,y) \ \leq \ R_{sens} \\ 0 & \text{for} \quad d_{ij}(x,y) \ > \ R_{sens} \end{cases} \tag{7.1}$$

The sensor field is represented by an mxn grid. $d_{ij}(x,y)$ denotes the Euclidean distance between a sensor node at (x,y) and any grid point at (i,j). The distances are measured in units of grid points. Equation 7.1 expresses the coverage $C_{ij}(x,y)$ of a grid point at (i,j) by a sensor at (x,y). The coverage for the entire grid is calculated as the fraction of grid points covered. In reality, sensor measurements are imprecise; hence the coverage needs to be expressed in probabilistic terms. In the stochastic sensor model shown in Figure 7.2, the probability of detection follows an exponential decaying function of distance from the sensor. The stochastic sensor model given in Equation 7.2 is motivated in part by [42],

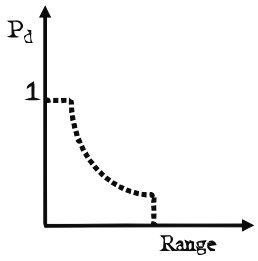


Figure 7.2: Stochastic sensor coverage model

$$C_{ij}(x,y) = \begin{cases} 1 & \text{for} \quad d_{ij}(x,y) \ \leq \ (R_{sens} - R_e) \\ e^{(-\lambda a^{\beta})} & \text{for} \quad (R_{sens} + R_e) \ < d_{ij}(x,y) \ < R_{sens} \\ 0 & \text{for} \quad d_{ij}(x,y) \ \geq \ (R_{sens} + R_e) \end{cases} \tag{7.2}$$

$R_e(R_e < R_{sens})$ is a measure of the uncertainty in sensor measurement, $a = d_{ij}(x, y) - (R_{sens} - R_e)$, and λ and β are parameters that measure the detection probability when there is an uncertainty in sensor detection. The coverage for the entire sensor field is calculated as the fraction of grid points that exceeds the threshold C_{th}. So the first objective is maximization of coverage. This objective can be calculated by the following expression:

$$Max \qquad Coverage(f_1) = \frac{\bigcup_{i=1,\ldots,N} A_i}{A} \tag{7.3}$$

where A_i is the area covered by the ith node, N is the total number of nodes and A is the area of the region of interest.

Lifetime

The second objective considered is maximization of lifetime. Lifetime is defined as the time until one of the participating nodes run out of energy. This objective can be calculated by the subsequent expression:

$$Max \qquad Lifetime(f_2) = \frac{T_{failure}}{T_{max}} \tag{7.4}$$

where $T_{failure}$ is the maximum number of sensing cycles before failure of any node and T_{max} is the maximum number of possible sensing cycles. In every sensing cycle, the data from every node is routed to HECN through a route of minimum weight. To find this route, the outgoing edges of every node are weighted by the inverse of node's remaining energy and then Dijkstra algorithm [43] is used to find out the route with minimum weight. This calculation is repeated for subsequent sensing cycles until energy of at least one node is depleted. This gives the maximum number of sensing cycles before any node fails. The maximum number of possible sensing cycles can be obtained by taking ratio of total energy of any node and energy lost for one data transmission. This corresponds to a layout when all sensors are directly connected to HECN.

These two objectives are competing with each other. The coverage objective will try to spread out the nodes for maximizing coverage while resulting in high energy loss and small lifetime. The lifetime objective will try to arrange the nodes as close as possible to the HECN for reducing loss in energy which results in poor coverage.

7.4 Proposed MOPSO Algorithm

In order to construct a direct relationship between the problem domain and the PSO particles for this problem, every particle represents coordinates of N number of nodes. So each particle represents a network layout. The proposed MOPSO algorithm is composed of the following steps:

1. Randomly initialize the positions for all the particles.
2. For each particle DO
   WHILE the particle has not formed a connected network
   (a) Add random numbers in the [0 1] interval to the position of the particle.
   (b) Constrain the position so that it does not exceed the bound given by the grid size.
3. Initialize the velocity of each particle.
4. Evaluate the two objective values of each particle using Equation 7.3 and 7.4.
5. Store the positions representing non-dominated solutions in the elite archive.
6. Initialize the memory of each particle.
7. WHILE the maximal number of iterations has not yet been reached
   For each particle DO
   (a) Compute the velocity as described in Eq. (1).
   (b) Compute the new position by adding the velocity to the previous position using Eq. (2).
   (c) Constrain the position so that it does not exceed the bound given by the grid size.
   (d) WHILE the particle has not formed a connected network
      i. Add random numbers in the [0 1] interval to the position of the particle.
      ii. Constrain the position so that it does not exceed the bound given by the grid size.
   (e) Evaluate the objective values of the current position.

(f) Compare the new position with members in the archive.

(g) Update the elite archive by inserting all the currently non-dominated positions and eliminate any dominated locations from the archive.

(h) When the new position dominates the local best, replace the local best.

(i) Locate the archive member that dominates the fewest particles in this iteration as the global best.

(j) Increase the loop counter.

8. Return the archive as the non-dominated solution set.

7.5 Results and Discussion

The MOPSO algorithm starts with a "swarm" of particles randomly generated where each particle represents a network layout represented by sensor co-ordinates. The coverage and lifetime of the particles are then calculated. The archive containing non-dominated Pareto optimal set of solutions is developed according to the Pareto optimal dominance developed by Coello and Lechuge [44]. The position and velocity of particles are updated using the gbest obtained from the archive.

The input parameters taken for simulation are as follows: grid size = 10*10, number of nodes = 10, number of particles = 50, number of generations = 10, R_{sens} = 2, R_{comm} = 2, R_e = 1, λ = 0.5, β = 0.5, C_{th} = 0.7. For coverage calculation, two experiments are carried out with two types of models explained in section 7.3.2.

Finally a well-populated Pareto front is obtained, from the external archive, which gives a solution set of layouts for optimization. The Pareto front obtained for a binary sensor model is shown in Figure 7.3. Two Pareto optimal layouts are shown in Figure 7.4 and 7.5 to illustrate the variety of layouts available. The effect of different sensor models on the Pareto front is shown in Figure 7.6. The improvement in Pareto front with increase in number of generations of MOPSO algorithm is also shown in Figure 7.7.

The layout optimization is a highly non-linear problem because a small change in the position of a sensor can lead to a disconnected network. The MOPSO provides a set of pareto-optimal layouts from which the end-user can choose the layout depending on the trade-off between coverage and lifetime. The layout shown in Figure 7.4 is the layout with best coverage. For getting more coverage the particles spread out to minimize the overlapping region. This leads to a network where the sensors are far away from HECN. Many sensors transmit their own data as well as act as communication relay for other far

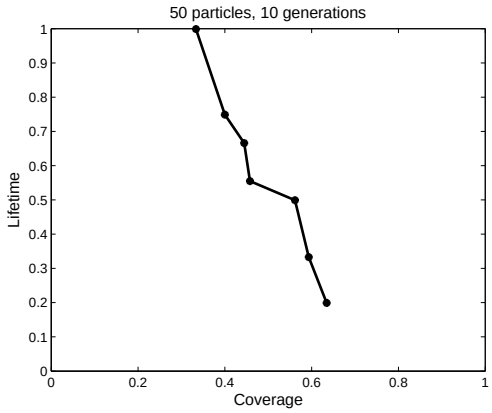


Figure 7.3: Pareto front for a WSN with 10 sensors

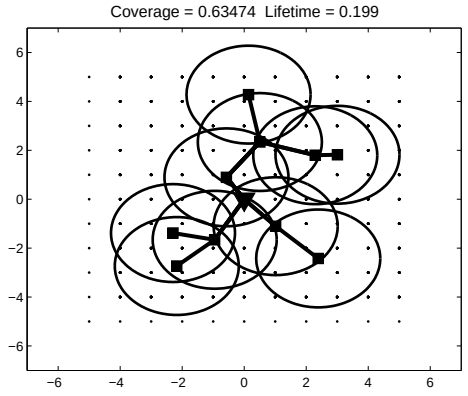


Figure 7.4: Pareto-optimal layout with best coverage for a WSN with 10 sensors, 50 particles , 10 generations

away sensors. Hence sensors acting as communication relay loss more energy. The layout shown in Figure 7.5 is example of another pareto optimal layout available to the user. It is more interesting to look at the Pareto fronts obtained using two different sensor models as shown in Figure 7.6. The upper bound for the coverage for the stochastic sensor model is lower than the upper bound for the case of binary sensor model. This is due to the fact that coverage for the binary sensor model is the fraction of the grid points covered by the circles. For the stochastic sensor model, even though there are a large number of grid points that are covered, the overall number of grid points with coverage probability greater than the threshold is fewer. Practically sensor measurement involves some uncertainty which is not considered in binary sensor model. Although stochastic sensor model gives less coverage, it is physically realizable. For a given number of sensors and total number of particles, the upper bound for coverage and lifetime

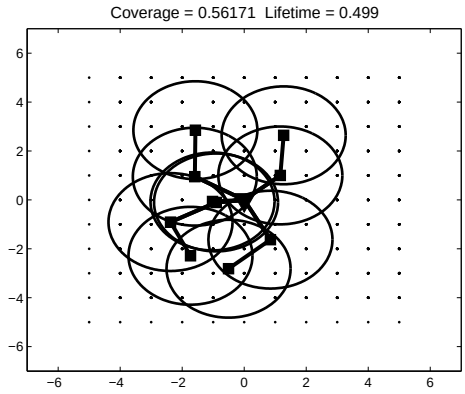


Figure 7.5: Example of another pareto-optimal layout for a WSN with 10 sensors, 50 particles , 10 generations

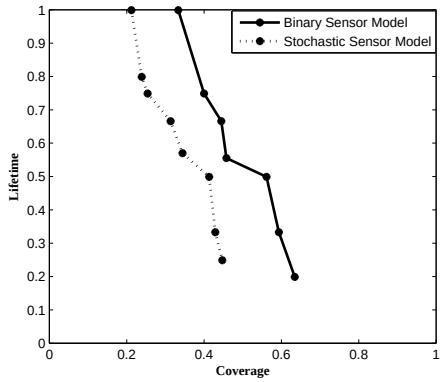


Figure 7.6: Pareto fronts obtained with different sensor models for a a WSN with 10 sensors, 50 particles , 10 generations

remains unchanged with the increase in the number of generations. The Pareto front shifts towards right with increase in generation resulting in a better set of Pareto-optimal solutions as shown in Figure 7.7. This improvement is achieved at the cost of increase in computation time. For those applications where computation time is not a limitation, MOPSO algorithm can be executed for large number of generations to obtain a better set of non-dominated solutions in the external archive.

7.6 Conclusion

Several papers have been reported in the literature showing the importance of wireless sensor network and the potential applications that are emerging with the development

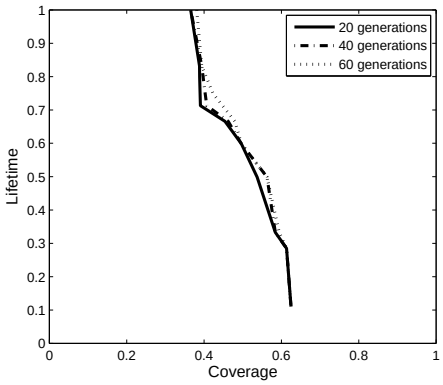


Figure 7.7: Pareto fronts obtained for a WSN with 10 sensors and 50 particles

of this new technology. How efficient layout can be obtained has been a challenging problem for the network designers. In this chapter, an energy efficient layout with good coverage for a WSN is considered. The application of Multi- Objective Particle Swarm Optimization to maximize coverage and lifetime simultaneously is discussed. Thus, the aim of the proposed algorithm is to locate good non-dominated solutions under time pressure. To optimize multiple objectives simultaneously, the proposed algorithm maintains an elite archive and uses the archive members to dynamically lead the particle swarm in searching for more and better non-dominated solutions. While applying the algorithm, the connectivity of the network is considered as a constraint. The mobility of the nodes provides a way to avoid time consuming, expensive layout. The end user is provided with a set of pareto-optimal solutions to choose from depending on the application. We also investigated the influence of number of iterations of the MOPSO algorithm on the Pareto front. The simulation results show that the binary sensor model gives better coverage than stochastic sensor model but it is difficult to realize practically. The efficiency of MOPSO algorithm will become apparent in more realistic conditions.

Chapter 8

Concluding Remarks and Future Work

8.1 Conclusion

Several efforts have been pursued in the literature to develop distributed estimation schemes based on consensus strategies. One of the main results of this work is to show that cooperation improves performance from the estimation point of view, in terms of saving computation and communication resources. Cooperation has a stabilizing effect on the network. One can design the individual filters using local information only in order to achieve (local) stability and implement incremental or diffusion protocols to improve global performance. Energy conservation arguments have been used to study the steady-state performance of the individual nodes for Gaussian data. Closed-form expressions for global and local mean and mean-square performance have been derived, matching very well the simulations that have been carried out. The inherent cooperative strategy of the incremental scheme not only improves performance, but it also decreases the amount of communication needed to implement cooperation among the nodes. The diffusion scheme results in peer-to-peer algorithms suitable for general topologies and robust to link and node failures. Besides robustness and spatial diversity, diffusion protocols improve the network estimation performance but with an additional level of complexity. For robust estimation of parameters, we have proposed a new cost function based on error saturation nonlinearity. Simulation results reveal that this scheme performs better estimation as compared to schemes using other robust norms like Huber loss function.

In this thesis we have also considered the deployment problem for mobile wireless sensor networks. A region of interest needs to be covered by a given number of nodes with limited sensing and communication range. We start with a "'random"' distribution of nodes over the region of interest. Though many scenarios adopt random deployment because of practical reasons such as deployment cost and time, random deployment may not provide a uniform distribution which is desirable for a longer system lifetime over the region of interest. In this thesis, we have proposed a multiobjective approach for the deployment of nodes to improve upon an irregular initial deployment of nodes. Coverage and lifetime are taken as the two conflicting objectives for achieving a set of layouts. Depending on the application, the user can choose a layout from the set of solutions. The performance of the MOPSO algorithm is determined by the computation time and uniformity of the solutions on the Pareto front. Simulation results show that the MOPSO algorithm obtains a better set of solutions as compared to single objective algorithms and other deployment algorithms. It provides a more uniform distribution from initial uneven distributions in an energy-efficient manner.

8.2 Scope for Future Work

More general data distribution, and also more sophisticated co-operative schemes with each node cooperating with a subset of nearby nodes, are useful extensions and will be studied in future work. We studied the LMS implementation operating with Gaussian signals. Other strategies can be studied using the formulation presented here, such as the distributed normalized LMS(dNLMS), the distributed affine projection algorithms(dAPA) and distributed RLS implementations. Analysis for these algorithms operating in networks with changing topologies and observing non-Gaussian data is available.

We investigated the steady-state performance of distributed incremental LMS algorithm with error saturation nonlinearity. This work can be extended to other family of error nonlinearities like LMF, Sign error *etc.* If both the desired and the input data are corrupted by impulsive noise, then the Huber Prior Error Feedback-Leat Square Lattice (H-PEF-LSL) algorithm [25] gives a good performance. Some robust algorithms are also available which are robust to changing topology, nodal failure and link failures.

We have discussed an energy efficient layout for wireless sensor network using MOPSO. In practice, a WSN is divided into multiple sub-regions for easy layout, organization and management. Since stochastic sensor model is accepted practically, the size of sub-region and their corresponding density and edge effects should be considered. In future work, we will also take energy consumption due to sensor movement into account. Other objectives, such as time and distance for sensor movement and uniformity of the network will also be considered for optimization.

Bibliography

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102–114, Aug 2002.

[2] D. Estin, R. Govindan, J. Heidemann, and S. Kumar, "Next century chanlanges:Scalable cordination in sensor network," in *Proc. ACM/IEEE MobiComm'99*, August 1999, pp. 263–270.

[3] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, July 2008.

[4] C. G. Lopes and A. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.

[5] B. Widrow and S. D. Strearns, *Adaptive Signal Processing.* Englewood Cliffs, NJ:Prentice-Hall, 1985.

[6] S. Haykin, *Adaptive filter theory.* Englewood Cliffs, NJ:Prentice-Hall, 2001.

[7] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM J. Optim*, vol. 7, pp. 913–926, 1997.

[8] A. Geary and D. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *Proceedings of the 38th IEEE Conference on Decision and Control*, vol. 1, pp. 907–912, 1999.

[9] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM J. on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[10] A. H. Sayed, *Fundamentals of Adaptive Filtering.* John Wiley and Sons. Inc. Publication, 2003.

[11] D. Estrin and M. Srivastav, "Instrumenting the world with wireless sensor networks," in *Int. conf. Acoustics, Speech, Signal Processing (ICASSP)*, May 2001, pp. 2033–2036.

[12] G. A. Clark, S. K. Mitra, and S. R. Parker, "Block implementation of adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 29, no. 3, pp. 744–752, June 1981.

[13] A. Feuer, "Performance analysis of the block least mean square algorithm," *IEEE Trans. Circuits Syst.*, vol. 32, no. 9, pp. 960–963, Sep. 1985.

[14] G. Panda, B. Mulgrew, and C. F. N. Cowan, "A self-orthogonalizing efficient block adaptibe filter," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 6, pp. 1573–1582, Dec. 1986.

[15] C. Burrus, "Block implementation of digital filters," *IEEE Trans. Circuit Theory*, vol. 18, no. 6, pp. 697–701, Nov. 1971.

[16] T. Y. Al-Naffouri and A. Sayed, "Transient analysis of data-normalized adaptive filters," *IEEE Trans. Signal Process.*, vol. 51, no. 3, pp. 639–652, March 2003.

[17] T. Al-Naffouri and A. Sayed, "Transient analysis of adaptive filters with error nonlinearities," *IEEE Trans. Signal Process.*, vol. 51, no. 3, pp. 653–663, March 2003.

[18] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd International Symposium on Information processing in sensor networks*, April 2004, pp. 20–27.

[19] S. C. Bang and S. Ann, "A robust adaptive algorithm and its performance analysis with contaminated-Gaussian noise," in *Proceedings of ISPACS*, Seol, Korea, Oct 1994, pp. 295–300.

[20] S. Koike, "Adaptive threshold nonlinear algorithm for adaptive filters with robustness against impulse noise," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 45, no. 9, pp. 2391–2395, Sep. 1997.

[21] V. Delouille, R. Neelamani, and R. G. Baraniuk, "Robust distributed estimation using the embedded subgraphs algorithm," *IEEE Trans. Signal Process.*, vol. 54, no. 8, pp. 2998–3010, AUG. 2006.

[22] N. J. Bershad, "On error saturation nonlinearities for LMS adaptation," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, no. 4, pp. 440–452, April 1988.

[23] N. Bershad, "On error saturation nonlinearities for LMS adaptation in impulsive noise," *IEEE Trans. Signal Process.*, vol. 56, no. 9, pp. 4526–4530, Sep. 2008.

[24] X. Wang and H. V. Poor, "Joint channel estimation and symbol detection in rayleigh flat-fading channels with impulsive noise," *IEEE Commun. Lett.*, vol. 1, no. 1, pp. 19–21, Jan. 1997.

[25] S. C. Chan and Y. X. Zou, "A recursive least m-estimate algorithm for robust adaptive filtering in impulsive noise: Fast algorithm and convergence performance analysis," *IEEE Trans. Signal Process.*, vol. 52, no. 4, pp. 975–991, April 2004.

[26] S. R. Kim and A. Efron, "Adaptive robust impulsive noise filtering," *IEEE Trans. Signal Process.*, vol. 43, no. 8, pp. 1855–1866, Aug. 1995.

[27] N. J. Bershad, "On weight update saturation nonlinearities in LMS adaptation," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 2, pp. 623–630, Feb. 1990.

[28] H. Fan and R. Vemuri, "Robust adaptive algorithms for active noise and vibration control," *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pp. 1137–1140 vol.2, Apr 1990.

[29] O. Abu-Ella and B. El-Jabu, "Optimal robust adaptive LMS algorithm without adaptation step-size," *Millimeter Waves, 2008. GSMM 2008. Global Symposium on*, pp. 249–251, April 2008.

[30] N. J. Bershad and M. Bonnet, "Saturation effects in LMS adaptive echo cancellation for binary data," *IEEE Trans. Signal Process.*, vol. 38, no. 10, pp. 1687–1696, Oct. 1990.

[31] T. I. Haweel and P. Clarkson, "A class of order stastistic LMS algorithms," *IEEE Trans. Signal Processing*, vol. 40, no. 1, pp. 44–53, Jan 1992.

[32] T. Y. Al-Naffouri and A. H. Sayed, "Adaptive filters with error nonlinearities: Mean-square analysis and optimum design," *EURASIP Journal on Applied Signal Processing*, pp. 192–205, Oct 2001.

[33] R. Price, "A useful theorem for non-linear devices having Gaussian inputs," *IEEE Trans. Inf. Theory*, vol. IT-4, pp. 69–72, June 1958.

[34] R. Pawula, "A modified version of price's theorem," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 285–288, Apr 1967.

[35] T. Al-Naffouri, A. Zerguine, and M. Bettayeb, "Convergence analysis of the LMS algorithm with a general error nonlinearity and an iid input," *Signals, Systems and Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on*, vol. 1, pp. 556–559, Nov 1998.

[36] S. Dhillon, K. Chakrabarty, and S. Iyengar, "Sensor placement for grid coverage under imprecise detections," in *Proceedings of Fifth International Conference on Information Fusion*, vol. 2, Seol, Korea, 2002, pp. 1581–1587.

[37] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. Saluja, "Sensor deployment strategy for detection of targets traversing a region," *Mobile Networks and Applications*, vol. 8, no. 4, pp. 453–461, 2003.

[38] K. Kar and S. Banerjee, "Node placement for connected coverage in sensor networks," in *Proceedings of WiOpt: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2003.

[39] K. Ferentinos and T. Tsiligiridis, "Adaptive design optimization of wireless sensor networks using genetic algorithms," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 51, no. 4, pp. 1031–1051, 2007.

[40] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor networks," in *Proceedings of Fifth International Conference on Commun.*, vol. 2, 2001, pp. 472–476.

[41] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *Proceedings of IEEE INFOCOM Conf.*, vol. 2, 2003, pp. 1293–1303.

[42] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 3, pp. 2349–2365, 1987.

[43] T. H. C. et al., *Introduction to algorithms*. Cambridge, MA: MIT Press, 2001.

[44] C. A. Coello, G. T. Pulido, and M. S. Lechuge, "Handling multiple objective with particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, pp. 256–279, Jun 2004.